

11.06.03

日 本 国 特 許 庁
JAPAN PATENT OFFICE

REC'D 01 AUG 2003

WIP PAT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: 2002年 6月12日

出 願 番 号
Application Number: 特願2002-171338
[ST. 10/C]: [JP2002-171338]

出 願 人
Applicant(s): 松下電器産業株式会社

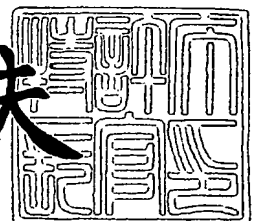
PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

CERTIFIED COPY OF
PRIORITY DOCUMENT

2003年 7月11日

特許庁長官
Commissioner,
Japan Patent Office

今 井 康 夫



【書類名】 特許願

【整理番号】 2117520293

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46 340

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

【氏名】 片岡 充照

【特許出願人】

【識別番号】 000005821

【氏名又は名称】 松下電器産業株式会社

【代理人】

【識別番号】 100098291

【弁理士】

【氏名又は名称】 小笠原 史朗

【手数料の表示】

【予納台帳番号】 035367

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9405386

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 サービス安全拡張プラットフォーム

【特許請求の範囲】

【請求項 1】 サービスと実行形式とが対応付けられており、前記実行形式の変更や追加によって前記サービスの拡張が達成されるサービス安全拡張プラットフォームであって、前記サービスの拡張を行うサービス依存 A P I を具備し、かつ前記実行形式からの前記サービスの拡張は前記サービス依存 A P I の呼び出しによってのみ行われることを特徴とする、サービス安全拡張プラットフォーム。

【請求項 2】 前記サービスの拡張が新規サービスの新設であることを特徴とする、請求項 1 に記載のサービス安全拡張プラットフォーム。

【請求項 3】 前記サービスの拡張がサービス利用開始であることを特徴とする、請求項 1 に記載のサービス安全拡張プラットフォーム。

【請求項 4】 注目する前記サービスの拡張は、注目する前記サービスに対応付けられた実行形式からの前記サービス依存 A P I の呼び出しによってのみ行われることを特徴とする、請求項 1 乃至請求項 3 に記載のサービス安全拡張プラットフォーム。

【請求項 5】 複数の前記サービス間に親子関係が定義され、前記実行形式が要求する前記サービス依存 A P I 呼び出しが前記サービスに対応付けられるサービス依存リソースを処理対象として指定した際に、前記実行形式に対応付けられるサービスが前記サービスの先祖である場合にのみ、サービス依存リソースに対して処理可能であることを特徴とする、請求項 1 乃至請求項 3 に記載のサービス安全拡張プラットフォーム。

【請求項 6】 メタサービスに対応付けられた前記実行形式が前記サービス依存 A P I によって、前記サービスの少なくとも 1 つが拡張可能なことを特徴とする、請求項 1 乃至請求項 3 に記載のサービス安全拡張プラットフォーム。

【請求項 7】 前記メタサービスに対応付けられた前記実行形式が前記サービス依存 A P I によって、前記サービスの全てが拡張可能であり、前記メタサービスに対応付けられていない前記実行形式は前記サービス依存 A P I によって前

記サービスの拡張が不可能であることを特徴とする、請求項 6 に記載のサービス安全拡張プラットフォーム。

【請求項 8】 前記実行形式がコンテンツとして満たすべき条件を満たした制御コンテンツであり、前記制御コンテンツが、前記コンテンツの少なくとも 1 つと共にコンテンツとして伝送され、前記コンテンツの少なくとも 1 つから前記制御コンテンツを指定する情報が伝送され、前記制御コンテンツによってのみサービス依存 API の処理が可能であることを特徴とする請求項 1 乃至請求項 3 に記載のサービス安全拡張プラットフォーム。

【請求項 9】 前記サービス依存 API によって、特定の前記サービスのコンテンツの自動的な格納が制御されることを特徴とする請求項 8 に記載のサービス安全拡張プラットフォーム。

【請求項 10】 前記実行形式を少なくとも 1 つのサービス提供部から送出し、前記実行形式を実行する少なくとも 1 つの端末で受信することを特徴とする、請求項 1 乃至請求項 9 に記載のサービス安全拡張プラットフォーム。

【請求項 11】 請求項 1 乃至請求項 10 に記載のサービス安全拡張プラットフォームを実現するサービス安全拡張方法。

【請求項 12】 請求項 1 乃至請求項 10 に記載のサービス安全拡張プラットフォーム実施するコンピュータプログラムを格納した記憶媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

デジタルコンテンツの提供サービスを実現する実行形式による不正アクセスの排除を、実行形式に不正アクセスを引き起こす処理が含まれることを否定する必要無く達成するという、新次元の安全性を実現した、サービスの拡張可能な全く新しい「サービス安全拡張プラットフォーム」に関する。詳述すれば、実行形式の配布元の確からしさからの推測を必要とせずに、他のサービスの状態変更、データ破壊、およびプラットフォーム自体のシステムダウンなどに代表される不正アクセスを排除できると共に、実行形式の変更や追加によってサービスの機能／諸元の変更や新規サービスの追加というサービス拡張も達成できるサービス安全

拡張プラットフォームに関する。

【0002】

【従来の技術】

上述の如く、本発明が提供するものは、従来にない全く新しく提供する「サービス安全プラットフォーム」であるので、従来の技術として示すべき適切な例を挙げるのは非常に困難である。それゆえに、先ず、本発明が初めて提唱すると信じるサービス安全拡張プラットフォームが固有に備える3つの主な特徴について以下に述べる。第1の特徴はサービスの拡張性であり、第2の特徴は安全性の確保であり、そして第3の特徴はサービス拡張操作のシームレス化である。

【0003】

第1の特徴である「サービスの拡張性」とは、サービス毎に異なるユーザインタフェースを持ち、ユーザインタフェースの変更や予期せぬサービスの追加が任意の時点で可能であることを言う。第2の特徴である「安全性の確保」とは、実行形式に不正アクセスを引き起こす処理が含まれることの否定を必要とせずに安全性が確保されることを言う。そして、第3の特徴である「サービス拡張操作のシームレス化」とは、サービス拡張のための操作手順が、サービスの利用時と同じ操作感のもとに行えることを言う。

【0004】

上述のように、これら3つの主な特徴を全て満たす「サービス安全プラットフォーム」を、従来技術において見いだすことはできない。しかしながら、強いて言えば、第1の特徴のみを満たすものとして、パーソナルコンピュータを利用してインターネットによるプッシュ型サービスを従来技術の一例として挙げる事ができる。ポイントキャストネットワーク社のポイントキャスト(R)は、ニュース配信サービスを実現したプッシュ型サービスである。パーソナルコンピュータを利用する複数のプッシュ型サービスの利用は、機械語で記述されたサービス毎に実装したブラウザをパーソナルコンピュータへのインストールすることに可能となる。

【0005】

ここで、プッシュ型サービスにおける、本発明の第1の特徴である「サービス

の拡張性」について説明する。プッシュ型サービスを実現するブラウザをパーソナルコンピュータのハードディスクにインストールすることで新たなサービスを追加することができる。ブラウザは機械語で記述されたコンピュータプログラムであり、サービス毎に固有のそれぞれ異なるユーザインタフェースを提供する。

【0006】

ブラウザのインストールは、従来、以下の様にして行われる。例えば、ブラウザをインストールする際には、先ず、ファイル転送プロトコル（例：ftp）のクライアントを起動しておき、ブラウザをクライアントのハードディスクなどにダウンロードし、次にダウンロードしたブラウザを起動することで実現される。また、新たなブラウザを利用する前には、ブラウザの設定を、ブラウザに組み込まれた設定メニューで操作して実現する。

【0007】

次に、本発明の第2の特徴である「安全性の確保」に関しては、従来は単なる憶測に基づくものであって、具体的な方策の裏付けを有するものではない。つまり、従来、ブラウザが安全であるということは、ブラウザが公式ホームページなど正式な配布先より入手したものであるもので、悪意のある第3者がコンピュータウィルスなどを感染させてないはずだという単なる願望に過ぎない。

【0008】

そして、サービスによる高度な提示機能を意味する本発明の第3の特徴である「サービス拡張操作のシームレス化」に関しては、従来の技術においては、サービス毎にブラウザを変更することで異なるユーザインタフェースを実現する。結果、操作のシームレス化の一部を構成するために必要な、複数サービスに対し異なるユーザインタフェースを持つ要件は満たされる。

【0009】

図18を参照して、上述のようなサービスの拡張は可能であるが安全ではない、従来のサービスプラットフォームについて具体的に説明する。同図に示すように、従来のサービスプラットフォームPFcは、サービス提供源1610、デリバリシステム120、および端末1630を含む。サービス提供源1610は、実行ファイル提供器1111およびファイル転送サーバ1611を含む。実行

ファイル提供器1111は実行ファイルを格納したハードディスク装置でよい。また、ファイル転送サーバ1611はWebサーバでよい。なお、実行ファイルとは、図19を参照して後述するように、端末1630のOSに直接渡して、実行されるプログラムファイルに代表されるバイナリデータである。

【0010】

デリバリシステム120は、サービス提供源1610から送られる実行ファイルを空間的または時間的にはなれた端末1630に向けて伝送する。

【0011】

端末1630は、エグゼキュータ1133および実行ファイル格納器1132を含む。エグゼキュータ1133は、実行ファイル格納器1132に格納された実行ファイルを起動することで端末1630におけるあらゆる処理を実行する。例えば、実行ファイルとしてファイル転送プログラムを起動した場合には、デリバリシステム120を経由して受信した、新たなサービスを実現する実行ファイルを実行ファイル格納器1132に格納させる。また、サービスを実現する実行ファイルを実行させれば、サービスをユーザに提供できる。

【0012】

図19に、サービスプラットフォームPFcにおいて、実行形式格納器132に格納されるデータを例示する。サービスS1の実行ファイルであるFE(S1)およびサービスS2の実行ファイルであるFE(S2)に並んで、実行ファイルの起動などのプラットフォーム操作を行うユーザインタフェースを提供するshellの実行ファイルFE(shell)や、新たなサービスを実現する実行ファイルFEをデリバリシステム120経由で端末1630に導入するファイル転送プログラムであるFE(ファイル転送)が実行ファイル格納器132内に格納されている。これらの実行ファイルは何れも機械語で実装される。

【0013】

【発明が解決しようとする課題】

図20に、サービスプラットフォームPFcのソフトウェア階層を示す。同図から読みとれるように、サービスプラットフォームPFcにおいて、ファイル転送実行ファイルを起動することで、あらゆるサービスを実現する実行ファイルを

新たに導入できる。このように、OS (Operating System) が構成するOS層の直上に、実行ファイルによってアプリケーション層が構成されている。結果、サービスを実現する実行ファイルが直接OSのリソースを参照したり操作できる。それゆえに、OSは、悪意を持った実行ファイルに対して無防備であるので、サービスプラットフォームPFCはとうてい安全とは言えない。

【0014】

また、本発明の第1の特徴である「サービスの拡張性」に関しては、新たなサービスの単純な導入は可能であるが、本発明が提供するコンテンツの視聴とシームレス化した新たなサービスの導入については、従来は不可能である。つまり、コンテンツの視聴と新たなサービスの導入とにおける操作性は、その見かけも手順も全く異なる。例えば、従来は、ブラウザをインストールする際には、ファイル転送プロトコルのクライアントを起動し、ブラウザをクライアントのハードディスクなどにダウンロードし、次にダウンロードしたブラウザを起動しなければならない。また、新たなブラウザを利用する前には、ブラウザの設定を、ブラウザによるコンテンツの視聴画面とは全く別の使い勝手の設定メニューを操作しなければならない。

【0015】

一方、本発明にかかるサービス安全拡張プラットフォームが提供するサービスの柔軟な拡張性は、ブラウザを終了したり、コンテンツの視聴と全く異なる使い勝手の手順を踏むことなく、あくまでもコンテンツの視聴と同じ操作の使い勝手の中でサービスの拡張を行えることである。すなわち、サービスの拡張のためだけの特別な使い勝手の操作をユーザに強要することなく、ユーザが最も慣れているであろうコンテンツの視聴と同じ使い勝手で、あたかも、通常の操作の延長として、サービスが拡張されていく状況を達成するものである。しかしながら、このように拡張性は、従来のサービスプラットフォームの提供できることではないことは上述の通りであり、そのために、全く新しいサービス拡張の実現方法が求められている。

【0016】

次に、本発明の第2の特徴である「安全性の確保」に関しても、正式な配布先

から入手したブラウザでもバグによって、他のプッシュ型サービスの動作に悪影響を与えたり、システム全体をダウンさせたりすることも少なくないのが現実である。そのため、従来においては、ブラウザのバイナリに正当性がある、すなわち、ブラウザのバイナリに不正アクセスの処理が含まれないであろうことを少なくとも保証する必要がある。しかし、その保証はあくまでも信頼関係に基づくものであって、ブラウザのバイナリに不正アクセス処理が含まれていないことを確実にするものではない。

【0017】

一方、本発明にかかるサービス安全拡張プラットフォームが提供する安全性は、如何なる実行形式（従来例でのブラウザに対応）の実行によっても、他のサービスへの悪影響や、システム全体のダウンを起こさない、全く新しい次元の安全性である。当然、従来の技術においては、このように方式は存在せず、それゆえに全く新しい実現方法が求められているものである。

【0018】

さらに、本発明の第3の特徴である高度な提示機能を実現する「サービス拡張操作のシームレス化」に関しても、従来の技術ではサービス管理とコンテンツの視聴とのシームレス化は不可能である。その原因は、サービス管理はブラウザの設定メニューで行われていることにある。つまり、サービスの管理に関わることは、コンテンツの視聴画面では行えないのである。一方、本発明にかかるサービス安全プラットフォームが提供するサービスによる高度な提示機能は、コンテンツにサービスの設定を含めることで、コンテンツ視聴と同一の使い勝手の中で、コンテンツの持つ高度な表現を持ち、かつ、サーバから自由に提示内容を設定することを可能とする。

【0019】

例えば、アンケートに記述しサーバに向けて回答を返信するといった、高度な記述能力を使用したコンテンツにサービスの設定機能を組み合わせることで、アンケートにユーザが興味にある内容を回答することで、自動的にコンテンツの取捨選択を行うためのユーザの嗜好情報が設定されるといった高度な提示機能を実現できる。また、ユーザにとっては、どこまでがコンテンツなのか、サービスの

設定なのかを全く意識させることなく、高度な設定を簡単な操作で行える。

当然、従来の技術においては、本発明にかかるサービス安全プラットフォームが提供するサービスによる高度な提示機能を実現する方式が存在せず、それゆえに全く新しい実現方法が求められているものである。

【0020】

よって、本発明は、上述の3つの特徴を実現するサービス安全プラットフォームを提供することを目的とする。

【0021】

【課題を解決するための手段および発明の効果】

第1の発明は、サービスと実行形式とが対応付けられており、実行形式の変更や追加によってサービスの拡張が達成されるサービス安全拡張プラットフォームであって、サービスの拡張を行うサービス依存APIを具備し、かつ実行形式からのサービスの拡張はサービス依存APIの呼び出しによってのみ行われることを特徴とする。

【0022】

上述のように、第1の発明においては、実行形式の正当性に依らない安全性が確保できる。

【0023】

第2の発明は、第1の発明において、サービスの拡張が新規サービスの新設であることを特徴とする。

【0024】

第3の発明は、第1の発明において、サービスの拡張がサービス利用開始であることを特徴とする。

【0025】

第4の発明は、第1の発明、第2の発明、および第3の発明の何れかにおいて、注目するサービスの拡張は、注目するサービスに対応付けられた実行形式からのサービス依存APIの呼び出しによってのみ行われることを特徴とする。

【0026】

第5の発明は、第1の発明、第2の発明、および第3の発明の何れかにおいて

、複数のサービス間に親子関係が定義され、実行形式が要求するサービス依存API呼び出しがサービスに対応付けられるサービス依存リソースを処理対象として指定した際に、実行形式に対応付けられるサービスがサービスの先祖である場合にのみ、サービス依存リソースに対して処理可能であることを特徴とする。

【0027】

第6の発明は、第1の発明、第2の発明、および第3の発明の何れかにおいて、メタサービスに対応付けられた実行形式がサービス依存APIによって、サービスの少なくとも1つが拡張可能なことを特徴とする。

【0028】

第7の発明は、第6の発明において、メタサービスに対応付けられた実行形式がサービス依存APIによって、サービスの全てが拡張可能であり、メタサービスに対応付けられていない実行形式(DE)はサービス依存APIによってサービスの拡張が不可能であることを特徴とする。

【0029】

第8の発明は、第1の発明、第2の発明、および第3の発明の何れかにおいて、実行形式がコンテンツとして満たすべき条件を満たした制御コンテンツであり、制御コンテンツ(DC)がコンテンツの少なくとも1つと共にコンテンツとして伝送され、コンテンツ(DC)の少なくとも1つから制御コンテンツを指定する情報が伝送され、制御コンテンツによってのみサービス依存APIの処理が可能であることを特徴とする。

【0030】

第9の発明は、第8の発明において、サービス依存APIによって、特定のサービスのコンテンツ(DC)の自動的な格納が制御されることを特徴とする。

【0031】

第10の発明は、第1の発明、第2の発明、第3の発明、第4の発明、第5の発明、第6の発明、第7の発明、第8の発明、および第9の発明の何れかにおいて、実行形式を少なくとも1つのサービス提供部送出し、実行形式を実行する少なくとも1つの端末で受信することを特徴とする。

【0032】

なお、本発明の一実施の形態においては、ブラウザを終了したり、コンテンツの視聴と全く異なる使い勝手の手順を踏むことなく、あくまでコンテンツの視聴と同じ操作の使い勝手の中でサービスの拡張を行える。すなわち、サービスの拡張のためだけの特別の使い勝手の操作をユーザに強要することなく、ユーザが最も慣れているであろうコンテンツの視聴と同じ使い勝手のうえで、いわば、いつの間にかサービスが拡張されている状況を達成できる。

また、如何なる実行形式（従来例でのブラウザに対応）の実行によっても、他のサービスへの悪影響や、システム全体のダウンを起こさない、全く新しい次元の安全性を達成できる。

【0033】

さらに、異なる実施の形態においては、コンテンツにサービスの設定を含めることで、コンテンツ視聴と同一の使い勝手の中で、コンテンツの持つ高度な表現を持ち、かつ、サーバから自由に提示内容を設定できる機能である。そして、ユーザにとっては、どこまでがコンテンツなのか、サービスの設定なのかを全く意識させることなく、高度な設定を簡単な操作で行える。

【0034】

【発明の実施の形態】

先ず、本発明にかかるサービス安全拡張プラットフォームの基本的概念について説明する。本発明にかかるサービス安全拡張プラットフォームにおいて、サービスは、従来のようなOSによって直接実行される、例えばバイナリの実行ファイルではなく、OSに対する実行を指示するエグゼキュータにより解釈されるいわばスクリプトとして構成される実行形式として構成される。そして、実行形式は、API呼び出しを含み、所定のサービスに対応付けられる。

【0035】

APIは、呼び出し時に、対応付けられたサービスに依存した挙動をするサービス依存である。サービス依存APIは、含まれる実行形式に対応付けられたサービスに固有のリソースに対して処理する。また、実行形式からリソースへのあらゆる処理はAPI経由に限定され、かつ、サービス依存APIにより処理されるリソースに対して処理を行うサービス依存API以外のAPIは存在しない。

また、実行形式内に記述するAPIのセットは、全てのサービスに対し共通に予め用意されたものであり、同一である。

【0036】

このように構成することによって、従来における実行ファイル（あるいは本発明における実行形式）の正当性に依らない新次元の安全性を確保し、サービスの拡張が可能な全く新しい「サービス安全拡張プラットフォーム」を創出する。つまり、本発明においては、ブラウザを終了したり、コンテンツの視聴と全く異なる使い勝手の手順を踏むことなく、あくまでもコンテンツの視聴と同じ操作の使い勝手の中でサービスの拡張が行えることを保証する。つまり、サービスの拡張のためだけに用意される特別の使い勝手の操作をユーザに強要することなく、ユーザが最も慣れているであろうコンテンツの視聴と同じ使い勝手で、ユーザがそれと特に意識することなく、気が付けけばサービスが拡張されている環境を提供する。また、如何なる実行形式（従来例でのブラウザに対応）の実行によっても、他のサービスへの悪影響や、システム全体のダウンを起こさない、全く新しい次元の安全性が確保できる。

【0037】

（第1の実施の形態）

図1、図2、図3、図4、図5、図6および図7を参照して、本発明の第1の実施の形態にかかるサービス安全拡張プラットフォームについて説明する。図1に示すように、本実施の形態にかかるサービス安全拡張プラットフォームSEP1は、サービス提供源110、デリバリシステム120、および端末130を含む。

【0038】

サービス提供部110は、端末部130が実行すべき新たなサービスを実現するための実行形式を送出する。デリバリシステム120は、サービス提供源110が送出的る情報を端末130に向けて時間的かつ／あるいは空間的に移動させる。デリバリシステム120は、インターネット通信網、放送や通信の無線ネットワーク、あるいは、DVD-ROM (Digital Versatile Disk-Read Only Memory) などのパッケージメディアと物

理的な物流システムの組み合わせで構成できる。

【0039】

端末130は、デリバリシステム120経由で受け取った情報を用いて、サービス提供源110から提供されたサービスを実行して、実行結果をユーザに提供する。なお、図1においては、簡便化のために、サービス提供源110、デリバリシステム120、および端末130のそれぞれの台数比が1:1:1であるように表されている。しかしながら、台数比は放送の形態に類似した1:1:c (cは任意の自然数) や、インターネットなどのa:1:c (aは任意の自然数) や、あるいは複数のデリバリシステムを持つ場合のa:b:c (bは任意の自然数) であってもよい。このような、一般化は以下に述べる全ての実施の形態において当てはまる。

【0040】

サービス提供源110は、実行形式提供器111、サービス識別設定器112、および送出器113を含む。実行形式提供器111は、サービスを実現する実行形式を格納し、必要に応じて出力する。この実行形式は、インターネットで用いられるHTML言語 (HyperText Markup Language) や、日本のデジタル放送のデータ放送で用いられるBML言語 (Broadcast Markup Language) などの、SGML (Standard Generalized Markup Language) /XML (Extensible Markup Language) 系のマークアップ言語や、仮想マシン上で動作するJava (R) 言語、あるいは機械語などでよい。ただし、実行形式は、後述する端末130に備えられたエグゼキュータを介して、端末130のOSに対して渡されて実行されるように構成される。

【0041】

サービス識別設定器112は、実行形式提供器111の出力する実行形式DEに対して、実行形式DEの属性等の副次的な事項を表す付随情報ISを生成すると共に実行形式DEに付与して識別化実行形式DE_iを生成する。付随情報ISは、対応する実行形式DEが実現するサービスとを対応付ける情報であるサービス識別情報Esと、その実行形式DEの使用条件、使用状態、および対応するサ

ービス内容などの情報を表す副次情報 α を含む。サービス識別情報 E_s は、例えば、重複しない数字であるIDや名称などでよい。副次情報 α はテキストでもコードでもよい。また、識別化実行形式 DE_i は、実行形式 DE と付随情報 IS を一体的に生成してもよいし、互いに独立して生成してもよい。本実施の形態においては、視認性を考慮して付随情報 IS は独立して生成される場合を例に説明する。

【0042】

送出器113は、サービス識別設定器112から入力される識別化実行形式 DE_i （実行形式 DE および付随情報 IS ）をデリバリシステム120に送出する。この送出を実現する伝送モデルは、いわゆるプル型およびプッシュ型の何れでもよい。

【0043】

プル型とは、インターネットのホームページ閲覧に用いられる伝送プロトコルであるHTTP（HyperText Transport Protocol）などで見られるように、受信側である端末130からの要求（demand）に基づき送出する伝送モデルである。またプッシュ型とは、デジタル放送の伝送に用いられる伝送プロトコルであるDSM-CC（Digital Storage Media Command & Control）データカーセルなどで見られるように、受信側の要求に関わらず所定のタイミングで送出側から送出する伝送モデルである。

【0044】

端末130は、ダウンローダ131、実行形式格納器132、エグゼキュータ133、リソースセクタ134、一般リソース管理者135、およびサービス依存リソース管理者136を含む。ダウンローダ131は、デリバリシステム120から伝送されてくる、識別化実行形式 DE_i （実行形式 DE および付随情報 IS ）を受信し、受信した識別化実行形式 DE_i を実行形式格納器132に書き込むと共に、識別化実行形式 DE_i から付随情報 IS を抽出してサービス依存リソース管理者136に出力する。付随情報 IS は、上述のようにサービス識別設定器112で設定されたサービス識別情報 E_s を含む。ダウンローダ131は、

送出器113とデリバリシステム120で実現される伝送モデルに整合しており、プッシュ型でもプル型でも実施可能である。

【0045】

識別化実行形式格納器132は、ダウンローダ131により書き込まれた識別化実行形式DE_i（実行形式DEおよび付随情報IS）を格納する。また、識別化実行形式格納器132は、要求に応じて格納した識別化実行形式DE_iをエクセキュータ133に出力する。識別化実行形式格納器132はハードディスクドライブ（HDD）やDVD-RAMなどの記録媒体や、フラッシュメモリやRAMなどの半導体メモリを用いて構成できる。

【0046】

図2に、本実施の形態において、識別化実行形式DE_iが識別化実行形式格納器132に格納される様子を示す。同例において、サービスS₁、サービスS₂乃至サービスS_n（ n は任意の自然数）に関して、識別化実行形式DE_i（S₁）はサービスS₁に対応し、識別化実行形式DE_i（S₂）はサービスS₂に対応し、識別化実行形式DE_i（S_n）はサービスS_nに対応する。

【0047】

そして、識別化実行形式DE_i（S₁）は、それぞれサービスS₁に対応するサービス識別情報E_s（S₁）と副次情報 α ₁から成る付随情報IS（S₁）と実行形式DEを含む。具体的には、サービス識別情報E_s（S₁）が実行形式DEとサービスS₁とを対応付けている。同様に、識別化実行形式DE_i（S₂）は、それぞれサービスS₂に対応するサービス識別情報E_s（S₂）と副次情報 α ₂から成る付随情報IS（S₂）と実行形式DEを含む。さらに、識別化実行形式DE_i（S_n）は、サービスS_nに対応するサービス識別情報E_s（S_n）と副次情報 α _nから成る付随情報IS（S_n）と実行形式DEを含む。実行形式DEはサービス識別情報E_s（S_n）によってサービスS₂に対応付けられている。

【0048】

サービスを個々に区別する必要のない場合、識別化実行形式DE_iはサービス識別情報E_sおよび副次情報 α から成る付随情報ISと実行形式DEで構成され

ると表現する。そして、サービスを個々に区別する必要がある場合は、識別化実行形式DE_i (S_o)は、サービス識別情報E_s (S_o)および副次情報 α_o から成る付随情報I_S (S_o)と実行形式DEで構成されると表現する。なお、 o は n 以下の任意の自然数である。

【0049】

図1に戻って、エグゼキュータ133は、識別化実行形式格納器132から入力される識別化実行形式DE_iに含まれる実行形式DEを解釈してサービスを実行する。ただし、識別化実行形式格納器132に格納された識別化実行形式DE_iに含まれる実行形式DEは、エグゼキュータ133以外では端末130内で解釈実行されることはない特徴を有していることは上述の通りである。エグゼキュータ133は、また、ユーザからのキーボードやポインティングデバイスや音声入力デバイスといった入力デバイスからの入力I_uや、GUIの画面表示や音声出力などの出力デバイスへの出力、といった対話処理を実行形式DEに基づいた手順で実現する。

【0050】

エグゼキュータ133は、実行形式DEがJava (R) 言語やマークアップ言語であれば仮想マシンや、一般にはブラウザと呼ばれる実行環境でよい。また、実行形式DEが機械語であれば、それを実行させるためのOSに付属するライブラリなどのミドルウェア群などである。

【0051】

エグゼキュータ133は、実行形式DEを解釈実行していく際に、実行形式DEにAPI (Application Program Interface) 呼び出しCapiが含まれている場合には、リソースセクタ134を経由して一般リソース管理者135もしくはサービス依存リソース管理者136に対してAPI呼出Capiを発行して、処理の実行を要求する。また、エグゼキュータ133は、現在実行している実行形式DEに対応付けられたサービス識別情報E_sをサービス依存リソース管理者136に通知する。

【0052】

リソースセクタ134は、エグゼキュータ133から発行されたAPI呼出

C a p i に基づいて、一般リソース管理者 135 およびサービス依存リソース管理者 136 の何れに対して、A P I 処理が要求されているのかを判断する。そして、要求されていると判断されている方に、A P I 呼出 C a p i を伝達する。なお、A P I 呼出 C a p i には、一般リソース管理者 135 が処理すべき一般の A P I（以後、「一般 A P I」と呼ぶ）と、サービス依存リソース管理者 136 が処理すべき A P I（以降、「サービス依存 A P I」）とに分類される。

【0053】

リソースとは、端末 130 がエグゼキュータ 133 を経由して、参照、変更、あるいは制御可能なあらゆる計算機資源を言う。すなわち、R A M への 1 次記録器や H D D などの 2 次記録器に格納されるデータ構造、制御可能な入出力デバイスに対するアクセス権と具体的な入出力制御、および通信制御などが含まれる。これらリソースのうち、他のサービスへの影響を与えることなく処理可能なリソースを一般リソースとし、サービス毎に存在し、かつサービスの拡張の際に処理が必要なリソースをサービス依存リソースと定義する。そして、一般リソースに対する A P I を一般 A P I とし、サービス依存リソースに対する A P I をサービス依存 A P I と定義する。

【0054】

さらに、2つの A P I と同一のリソースとの関係に例をおいて説明する。そして、一方の A P I はリソースを参照して他のサービスに影響しないが、他方の A P I はリソースを変更して他のサービスに影響を及ぼすと想定する。この場合、前者が一般 A P I に対応し、後者がサービス依存 A P I に対応する。しかしながら、説明の簡便化のために、参照のためのリソースと変更のためのリソースとがそれぞれ独立に存在すると見なして、前者と後者の内容が整合されていると捉えて説明する。

【0055】

サービス依存 A P I の具体例としては、サービスの利用状態を変更する関数（以後、「サービス利用状態操作関数」と称する）などが考えられる。サービス利用状態操作関数によって、利用者が端末 130 で個別のサービスを利用するのかしないのかを指定し、サービスを提供する際に端末 130 が行うべき処理の起動

と停止などを制御する。具体的には実行形式DEあるいは識別化実行形式DE i の内部データの初期化や必要な情報の受信などの前処理や、デリバリシステムを経由したサービス提供部に対するサービス利用契約の締結、課金、およびユーザ登録などである。

【0056】

一方、一般APIの具体例としては、端末受信部130の画面表示やキーボード入力といった入出力デバイスへの操作や、RAMへの一時記憶やHDDへの2次記憶に対するデータの読み書きなどがある。ただし、一般APIのリソースである画面表示を例にした場合に、複数のサービスに対応する画面表示が同時に出現する際には、サービス間で競合が発生することも考えられる。しかしながら、本実施の形態においては、実際には同時には1つのサービスのみが画面を占有するという制約を一般リソース管理者135やAPIの呼び出し方法などで実現すれば、あらゆる実行形式DEに対しても実際には競合が発生しない。

【0057】

一般リソース管理者135は、一般リソースを格納および管理する。一般リソース管理者135は、さらに、リソースセクタ134から入力されるAPI呼出Capiに基づいて、一般リソースへの参照や操作などを行う。例えば、画面描画のAPIが呼び出されると画面描画を行う一般リソースであるグラフィック表示デバイスに対して命令を発する。

【0058】

サービス依存リソース管理者136は、サービス依存リソースを格納および管理する。リソースセクタ134から入力されるAPI呼出Capiに基づいて、サービス依存リソースRSへの参照や操作などを行う。また、ダウンロード131から入力される付随情報ISに基づいて、サービス依存リソースRSの管理処理を行う。

【0059】

図3を参照して、サービス依存リソース管理者136に格納されるサービス依存リソースRSを管理するために生成されるサービス依存リソース管理テーブルTrsについて説明する。同図に例示するように、サービス依存リソース管理テ

ーブル $T r s$ は、少なくとも、 n 種類のサービス $S 1 \sim S n$ を表す複数の行 $L 1 \sim L n$ と、サービス S 毎の利用状態を表す 2 列 $C 1$ および $C 2$ からマトリックス状に構成されるデータベースである。具体的には、同図において行 $L 1$ がサービス $S 1$ に対応し、行 $L 2$ がサービス $S 2$ に対応し、行 $L n$ がサービス $S n$ （この場合、 n は 3 以上の自然数）に対応している。そして、列 $C 1$ はサービス識別情報 $E s$ に対応し、列 $C 2$ は利用状態に対応する。

【0060】

なお、より具体的に言えば、列 $C 1$ および列 $C 2$ の値は、それぞれ図 2 に模式的に表したように、サービス識別情報 $E s (S n)$ の $S n$ および副次情報 αn に基づいて決定されて書き込まれる。図 3 に示す例においては、行 $L 1$ に示されるサービス $S 1$ は「未利用」であり、行 $L 2$ に示されるサービス $S 2$ は「利用」状態であり、行 $L n$ に示されるサービス $S n$ は「利用」状態であることが分かる。このように行 $L 1 \sim L n$ のそれぞれは、異なるサービス $S 1 \sim S n$ のサービスを識別する情報を蓄えるために設けられている。この意味において、行 $L 1 \sim L n$ （行 $L o$ ）をサービス識別行と呼ぶ。

【0061】

次に、新たなサービス識別行を追加する際の動作について説明する。ダウンロード 131 は、サービス（識別化実行形式 $D E i$ ）を受け取ると、識別化実行形式格納器 132 に格納されている実行形式 $D E$ が対応するサービス依存リソース $R S$ を規定するサービス識別行がサービス依存リソース管理テーブル $T r s$ に追加される。これらの処理によって、自動的にサービスが端末 130 に導入されることによって、新サービスがサービス依存リソース管理者 136 で登録・管理される。

【0062】

以下に、図 4 を参照して、サービス依存リソース管理者 136 による新サービスの登録管理ルーチンの動作について説明する。新たなサービスに対応する識別化実行形式 $D E i$ （実行形式 $D E$ および付随情報 $I S$ ）は、サービス提供源 110 から端末 130 に送出されて、先ずダウンロード 131 に入力される。ダウンロード 131 は、識別化実行形式 $D E i$ から付随情報 $I S$ を抽出して、サービス

依存リソース管理者136に出力する。そして、サービス依存リソース管理者136においては、付随情報ISが入力されて時点で、本ルーチンの動作が開始される。

【0063】

そのため、ステップS502において、付随情報ISがサービス依存リソース管理者136に入力されているか否かが判断される。入力されていない場合、Noと判断されて、本ステップにおける処理が繰り返される。一方Yesの場合、処理は次のステップS504に進む。つまり、サービス依存リソース管理者136は、ダウンロード131から付随情報ISを受け取るまではサインサービスの登録管理処理は、実質的に開始されない。

【0064】

ステップS504において、ステップS502で受け取った付随情報ISのサービス識別情報Esが示すサービスSが、サービス依存リソース管理者136に格納されているサービス依存リソース管理テーブルTrsに既に登録されているか否かが判断される。含まれない、つまり、未登録サービスである場合は、Noと判断されて、処理は次のステップS506に進む。

【0065】

ステップS506において、サービス依存リソース管理者136において、サービス依存リソース管理テーブルTrsに新サービスに対応する新サービス識別行(Ln+1)が追記される。以降、図3に示すサービス依存リソース管理テーブルTrsを例として説明する。サービス依存リソース管理テーブルTrsには、既にサービスS1～Snまでが登録されているので、新たなサービスS(n+1)を登録するために、サービス識別行L(n+1)が追加される。そして、処理は次のステップS508に生成される。

【0066】

ステップS508において、新たに受領した付随情報ISに含まれるサービス識別情報Esに基づいて、行L(n+1)列C1にサービスS(n+1)を識別する値であるS(n+1)が記入される。そして、処理は次のステップS510に進む。

【0067】

ステップS514において、サービス依存リソース管理テーブルT r sに追加された行L (n+1) 列C2の利用状態の欄に「未利用」が記入される。これは、新規サービス（識別化実行形式D E i）が受信された際の初期状態としては、「未利用」を設定するようにしているからである。しかしながら、初期状態が「利用」であったり、一定期間の間デモとして試用するなどの設定にしてもよいし、また、これらの初期状態の何れを採るか示す情報を、付随情報I S（副次情報 α ）としてサービス識別設定器112によってサービス毎に与えてもよい。本ステップの処理の終了後、本ルーチンを終了する。

【0068】

一方、上述のステップS504において、Y e s、つまり含まれると判定された場合、上述のステップS513およびステップS514をスキップして、本ルーチンにおける処理を終了する。すなわち、ダウンロード131が受け取ったサービスが既に導入されているサービスである場合には、サービス登録は不要であるので、本ルーチンの処理が直ちに終了される。

【0069】

次に、図5を参照して、端末130によるサービス実行ルーチンの動作について説明する。具体的には、端末130において、エグゼキュータ133が呼び出すA P Iに対する実行形式D Eが実行されることによって、サービス実行が実現される。つまり、エグゼキュータ133が識別化実行形式格納器132から入力される識別化実行形式D E iに含まれる実行形式D Eを実行させるために、A P I呼出C a p iを発行した時点で、本ルーチンにおける実質的処理が開始される。

【0070】

よって、ステップS512において、リソースセクタ134によって、エグゼキュータ133から発行されたA P I呼出C a p iに基づいて、呼び出されたA P Iがサービス依存A P Iであるか否かが判断される。サービス依存A P Iであれば、Y e sと判断されて、処理は次のステップS514に進む。

【0071】

ステップS514において、サービス依存リソース管理者136によって、現在実行している実行形式DEに対応付けられたサービスのサービス識別情報Esを、エグゼキュータ133から得る。そして、処理は、次のステップS516に進む。

【0072】

ステップS516において、サービス依存リソース管理者136によって、API呼出Capiが処理対象として指定するサービス依存リソースRSが、ステップS514で検出された現在実行中の実行形式DEに対応するサービスに対応するか否かが判断される。実行中の実行形式DEに対応するサービスに対応する場合は、Yesと判断されて処理は次のステップS518に進む。

【0073】

ステップS518において、サービス依存リソース管理者136によって、サービス依存リソースに対するサービス依存APIの処理が行なわれる。そして、本ルーチンにおける処理は終了される。

【0074】

一方、上述のステップS512において、No、つまりサービス依存APIではない（すなわち一般APIである）と判断される場合、処理はステップS520に進む。

【0075】

ステップS520において、一般リソース管理者135によって、一般リソースに対して処理が行われる。そして、本ルーチンにおける処理は終了される。

【0076】

さらに、上述のステップS516においてNo、つまり実行中の実行形式DEに対応するサービスに対応しない場合、処理はステップS521に進む。

【0077】

ステップS521において、エラー処理が行われた後に、本ルーチンが終了される。このように、実行中の実行形式DEが対応しているサービス（S512でYes）であっても、操作不可能（S516でNo）なサービス依存リソースに対するAPI処理を許可しないように設定している。

【0078】

言い換えれば、本実施の形態においては、実行中の実行形式DEに対応付けられたサービスに関するサービス依存リソースのみが操作可能と設定される。このため、実行形式DEによって如何なるAPI呼び出しを発行させても、サービス依存リソース管理器136は他のサービスに対する参照や操作を排除できる。つまり、他のサービスに対する参照や操作を行うようなサービスの実行形式DEが入力されても、ステップS516を経てステップS521でエラー処理が行われて、そのような参照や操作を防止すると共に、そのような要求を検出できる。

【0079】

上述のように、本実施の形態においては、ステップS512において、リソースセクタ134によって、サービス依存リソースに対する処理はサービス依存APIでのみ操作できる様に規制されている。これについて、図6に示すサービス安全拡張プラットフォームSEP1の端末130のソフトウェア階層を参照して説明する。

【0080】

図6に示すように、サービス安全拡張プラットフォームSEP1の端末130をソフトウェア構成から見れば、最下層に基本ソフトウェアであるOSにより実現されるOS層が存在する。そして、OS層の直上に、それぞれ一般リソース管理器135、サービス依存リソース管理器136、および識別化実行形式格納器132を機能させる一般リソース管理ソフトウェア、サービス依存リソース管理ソフトウェア、および識別化実行形式格納ソフトウェアを有する。そして、一般リソース管理ソフトウェアおよびサービス依存リソース管理ソフトウェアの直上にはリソースセクタ134を機能させるリソースセレクトソフトウェアを有する。これらの、一般リソース管理ソフトウェア、サービス依存リソース管理ソフトウェア、リソースセレクトソフトウェア、および識別化実行形式格納ソフトウェアは、ミドルウェア層を構成する。

【0081】

ミドルウェア層のリソースセレクト層の直上には、エグゼキュータ133を機能させるエグゼキュートソフトウェア（エグゼキュータ）が位置し、識別化実行

形式格納ソフトウェアの直上にはダウンロード 131 を機能させるダウンロードソフトウェア（ダウンロード）が位置している。そして、これらのエグゼキュートソフトウェアとダウンロードソフトウェアは共に、アプリケーション層を構成している。

【0082】

そして、アプリケーション層のエグゼキュートソフトウェアの直上には、各サービス S1 ～ Sn を実行するサービス S1 ～ Sn 実行形式が位置して、コンテンツ層を構成している。

【0083】

このように、図 6 に示されるソフトウェア構成から明らかなように、サービス安全拡張プラットフォーム SEP1 においては、サービスの実行形式 DE はリソースセクタ経由の API 呼び出したリソースを参照、あるいは操作したりできない。よって、サービス依存リソースに対しては、サービス依存 API を呼び出すことが必須であるので、サービス依存リソース管理者 136 を経由しなければ参照したり操作したりできない。

【0084】

次に、図 7 に示す、サービス安全拡張プラットフォーム SEP1 によってユーザに提示される画面例を参照して、新たに追加されたサービスを実際に利用する状態に変更する際の動作について簡単に説明する。図 7 には、実行形式 DE により表示される新規サービス利用開始の是非をユーザに問い合わせる画面の一例が示されている。

【0085】

画面 SM は、新たなサービスである「マイ・ニュース・サービス」を実現する実行形式 DE をエグゼキュータ 133 で実行することで提示される画面の 1 例である。画面 SM 上にはサービスの利用を宣言するボタン BY と、利用しないことを宣言するボタン BN が配置されている。ユーザは、入力デバイス进行操作してボタン BY を選択するとこのサービスの利用が開始される。

【0086】

ここでボタン BY が選択された場合の動作について説明する。ボタン BY には、

、サービスの利用を開始を宣言するサービス依存APIを起動する様に実行形式DE中にプログラミングされている。このためボタンBYが選択されるとリソースセクタ134を経由してサービス依存APIがサービス依存リソース管理器136に届き、サービス依存リソース管理テーブルTrsに記載されているサービス依存リソースRSに該当するサービスの利用状態の欄の値を「利用」に書き換える。

【0087】

なお、図7に示す例では、サービスの単純な利用開始について述べたが、サービス提供部に対するサービス利用契約の締結やユーザ登録、あるいは蓄積型のデータ放送における視聴前の事前の自動蓄積処理開始なども同様に実施可能である。

このように、サービスの使用／未使用の状態遷移など、サービスに関する操作をサービス自身の実行形式DEによってユーザとの対話を行うことが可能となる。さらに、如何なる悪意を持った実行形式DEを他のサービスが実装された場合にであっても、サービスやプラットフォームに対する、誤動作やハングアップに代表される、あらゆる悪影響を排除できる。

【0088】

上述のように、本発明の第1の実施の形態にかかるサービス安全拡張プラットフォームSEP1においては、実行形式DEに不正な処理を引き起こすコードが含まれていないことを実行形式DEの配布元の身元の確からしさから類推するような不確実な安全性ではなく、どの様な実行形式DEを想定しても他のサービスやプラットフォーム自体に対する不正な処理の影響を引き起こさない、完全な安全性を、サービスの拡張性を保ったままで確保できる。

【0089】

(第2の実施の形態)

以下に、図8、図9、および図11を参照して、本発明の第2の実施の形態にかかるサービス安全拡張プラットフォームについて説明する。図8に示すように、本実施の形態の形態にかかるサービス安全拡張プラットフォームSEP2は、図1に示すサービス安全拡張プラットフォームSEP1におけるサービス提供源

110がサービス提供源210に変更され、端末130が端末230に変更されている。

【0090】

サービス提供源210は、サービス提供源110におけるサービス識別設定器112がメタサービス指定器212に交換されている。端末230は端末130におけるエグゼキュータ133がエグゼキュータ233に交換されると共に、リソースセクタ134とサービス依存リソース管理者136との間にメタサービス判定器234が新たに設けられている。

【0091】

端末230は、第1の実施の形態における端末130のダウンローダ131がダウンローダ231に置き換えられ、識別化実行形式格納器132が識別化実行形式格納器232に置き換えられ、エグゼキュータ133がエグゼキュータ233に置き換えられると共に、リソースセクタ134とサービス依存リソース管理者136との間にメタサービス判定器234が新たに設けられている。

【0092】

サービス提供源210において、メタサービス指定器212は、上述のサービス識別設定器112にメタサービス指定情報生成機能が付与されている。つまり、メタサービス指定器212は、サービス識別設定器112と同様に、実行形式DEの属性等の副次的な事項を表す付随情報ISを生成すると共に、メタサービスに対応付ける実行形式DEを指定するメタサービス指定情報ISMを生成する。

【0093】

メタサービスとは、上述のサービスの一種であるが、実際にユーザの利用するサービスの利用開始／終了などのサービスの機能／諸元の変更や新規サービスの追加といったサービスの拡張を行うことができる唯一のサービスであり、サービスの拡張を目的として存在するものである。この意味において、本明細書においては上述の第1の実施の形態にかかるサービスS_oと、本実施の形態におけるメタサービスS_{meta}を区別して説明する。

【0094】

付随情報 I S が実行形式 D E に付与されて識別化実行形式 D E i が生成され、メタサービス指定情報 I S m が実行形式 D E に付与されてメタサービス実行形式 D E m が生成される。つまり、メタサービス指定器 2 1 2 からは、識別化実行形式 D E i とメタサービス実行形式 D E m が混在して出力される。そして、これらの識別化実行形式 D E i とメタサービス実行形式 D E m は、送出器 1 1 3 およびデリバリシステム 1 2 0 を介して端末 2 3 0 に入力される。そして、端末 2 3 0 のダウンロード 2 3 1 によって、識別化実行形式 D E i とメタサービス実行形式 D E m は識別化実行形式格納器 2 3 2 に出力される。そして、ダウンロード 2 3 1 は識別化実行形式 D E i から付随情報 I S を抽出し、メタサービス実行形式 D E m からメタサービス指定情報 I S m を抽出して、それぞれをサービス依存リソース管理者 2 3 6 に出力する。

【0095】

図 9 に、メタサービス指定器 2 1 2 から出力された識別化実行形式 D E i とメタサービス実行形式 D E m が識別化実行形式格納器 2 3 2 に格納される様子を示す。同図に示す例においては、図 2 に示した第 1 の実施の形態におけるのと同様に、サービス S 1、サービス S 2 乃至サービス S n (n は任意の自然数) に対応する識別化実行形式 D E i (S 1)、D E i (S 2)、および D E i (S n) が例示されている。ただし、本図においては、サービス S 3 に対応する識別化実行形式 D E i (S 3) の位置に、初めてのメタサービス実行形式 D E m (1) が表示されている。メタサービス実行形式 D E m (1) は、実行形式 D E に、それがメタサービスであることを示すメタサービス指定情報 I S m (1.) が付与されている。

【0096】

メタサービス実行形式 D E m (1) は、それがメタサービス m 1 に対応するサービス識別情報 E s (S m 1) と副次情報 α m 1 から成るメタサービス指定情報 I S m と実行形式 D E を含む。具体的には、サービス識別情報 E s (S m 1) が実行形式 D E とメタサービス 1 とを対応付けている。このようにサービス識別情報 E s (S m 1) の「S m 1」がメタサービス 1 を規定する点を除けば、メタサービス指定情報 I S m も基本的には付随情報 I S と同じものである。

【0097】

つまり、メタサービス S_m は、上述のように、サービス S_n の1つであるので、メタサービス実行形式 DE_m は、識別化実行形式 $DE_i(S_m)$ と表すこともできる。よって、メタサービス指定情報 IS_m も付随情報 IS と総称してもよいが、本明細書においては、本実施の形態における特徴をわかりやすくするために、メタサービス指定情報 IS_m を付随情報 IS と区別して説明する。さらに、メタサービスに対応するサービス識別情報 E_s を、メタサービス識別情報 E_{sm} と称して、実行形式 DE に対応するサービス識別情報 E_s と区別して説明する。

【0098】

識別化実行形式格納器232の動作は、上述の第1の実施の形態にかかる識別化実行形式格納器132と基本的に同じである。ただし、識別化実行形式格納器232は、識別化実行形式 DE_i とメタサービス実行形式 DE_m とを格納し、要求に応じて、識別化実行形式 DE_i あるいはメタサービス実行形式 DE_m をエグゼキュータ233に出力する。

【0099】

エグゼキュータ233は、識別化実行形式 DE_i からサービス識別情報 E_s を抽出し、メタサービス実行形式 DE_m からサービス識別情報 E_{sm} を抽出して、それぞれをメタサービス判定器234に出力する。

【0100】

メタサービス判定器234は、リソースセクタ134から入力されるAPI呼出 Cap_i と、エグゼキュータ133から入力されるサービス識別情報 E_s およびサービス識別情報 E_{sm} に基づいて、実行中の実行形式 DE がメタサービスに対応付けられている場合にのみ、API呼出 Cap_i をサービス依存リソース管理者136に出力する。

【0101】

サービス依存リソース管理者136は、サービス依存リソース RS を格納すると共にサービス依存リソース管理テーブル Tr_s によってサービス依存リソース RS を管理する。なお、サービス依存リソース管理者136に格納されるサービス依存リソース RS およびそのサービス依存リソース管理テーブル Tr_s は、図

3を参照して説明した第1の実施の形態におけるものと同じでよい。

【0102】

そして、サービス依存リソース管理者136は、メタサービス判定器234を経由して入力されるAPI呼出Capiに応答して、サービス依存リソース管理者136に格納されているサービス依存リソースRSへの参照や操作などを行う。また、ダウンローダ131から供給される付随情報ISに基づいて、実行形式DEのそれぞれが対応するサービスの内容を認識する。

【0103】

新たなサービスの追加は、サービス依存リソース管理者136が、ダウンローダ231から入力される付随情報ISおよびメタサービス指定情報ISMに基づいて、サービス依存リソース管理テーブルTrsに登録されていない新規のサービスの存在を検出した時点で、そのサービスに対する識別行を追加して、内容を記述する。つまり、サービス依存リソース管理テーブルTrsには、列C1にはメタサービスの内容が書き込まれるが、サービス識別行追加の手順だけに注目すれば、第1の実施の形態における処理内容と同様である。

【0104】

次に、図10に示すフローチャートを参照して、端末230によるサービス実行ルーチンの動作について説明する。具体的には、端末230において、エグゼキュータ233が呼び出すAPIに対する実行形式DEが実行されることによって、サービス実行が実現される。つまり、エグゼキュータ133が識別化実行形式格納器132から入力される識別化実行形式DEiおよびメタサービス実行形式DEmに含まれる実行形式DEを実行させるために、API呼出Capiを発行した時点で、本ルーチンにおける処理が開始される。

【0105】

なお、図10に示すフローチャートは、上述の図5に示すフローチャートにおいて、「実行中の実行形式DEのサービス識別情報Esを実行エンジンから取得」ステップS514が「実行中の実行形式DEのサービス識別情報Esまたはサービス識別情報Es mを取得」ステップS1014に交換され、「実行中のサービスに対する操作かを判定」ステップS516が「実行中のメタサービスに対す

る操作かを判定」ステップS1016に交換されている点を除いては同様に構成されている。以下、本実施の形態に固有の動作に重点をおいて説明する。

【0106】

よって、ステップS512において、リソースセクタ134によって、エグゼキュタ133から発行されたAPI呼出Capiに基づいて、呼び出されたAPIがサービス依存APIであると判断されて、処理はステップS1014に進む。

【0107】

ステップS1014において、メタサービス判定器234によって、エグゼキュタ233から現在実行中の実行形式DEに対するサービス識別情報Esあるいはメタサービス識別情報Es mが読み出される。そして、処理は次のステップS1006に進む。

【0108】

ステップS1016において、実行中の実行形式DEがメタサービスに対応付けられているか否かが判断される。Noの場合、上述のエラー処理ステップS521を経て、本ルーチンが終了される。一方、実行中の実行形式DEがメタサービスに対応付けられている場合は、Yesと判断されて、処理は上述の「サービス依存リソースに対して処理」ステップS518を経て本ルーチンが終了される。

【0109】

なお、ステップS518においては、実行中のメタサービスの実行形式DEであれば（ステップS1016でYes）、全てのサービスに対するサービス依存リソースに対する処理が実行される。

【0110】

上述のように、第2の実施の形態においては、実行形式DEがメタサービスであれば、全てのサービスに対するサービス依存APIが実行できる。従ってメタサービスの実行形式DEの画面上で、サービス提供部から提供可能なサービス一覧を表示し、一覧表示上で、サービスの機能／諸元の変更や新規サービスの追加に代表されるサービスの拡張が達成される。

【0111】

(第3の実施の形態)

次に、図11、図12、および図13を参照して、本発明の第3の実施の形態にかかるサービス安全拡張プラットフォームについて詳細に説明する。図11に示すように、本実施の形態にかかるサービス安全拡張プラットフォームSEP3は、図1に示すサービス安全拡張プラットフォームSEP1におけるサービス提供源110がサービス提供源310に変更され、端末130が端末330に変更されている。

【0112】

サービス提供源310は、サービス提供源110におけるサービス識別設定器112がサービス識別設定器312に交換されると共に、サービス識別親子管理器314が新たに設けられている。サービス識別親子管理器314は、個々のサービスに親子関係がある場合に、そのような関係を識別情報Esの間の親子関係として管理する。さらに、管理している親子関係を示すサービス親子指定情報Ishをサービス識別設定器312に出力する。

【0113】

端末330は、端末130におけるダウンローダ131がダウンローダ331に交換され、識別化実行形式格納器132が識別化実行形式格納器332に交換され、エグゼキュータ133がエグゼキュータ233に交換され、親子判定器334がリソースセクタ134とサービス依存リソース管理器136の間に新たに設けられている。

【0114】

サービス提供源310において、サービス識別親子管理器314が管理するサービスの親子関係とは、親サービスが子サービスのサービス依存リソースRsを操作可能とする関係と定義される。なお、必要に応じて、親サービスSp、子サービスSc、親サービスのサービス依存リソースRS cを称して、互いに識別するのとする。例えば、「音楽コンテンツ配信サービス」といったサービスのカテゴリに対応する親サービスに対して、「松下ミュージック配信サービス」や、「テイクミュージック配信サービス」、および「イーピーチャンネル音楽サー

ビス」に代表される個々のサービスの種類に対応する子サービスがある。

【0115】

サービス識別設定器312は、上述のサービス識別設定器112に親子サービス識別情報生成機能が付与されている。つまり、サービス識別設定器312は、サービス識別設定器112と同様に、実行形式DEの属性等の副次的な事項を表す付随情報ISを生成する。さらに、サービス識別設定器312は、サービス識別親子管理者314から供給されるサービス親子情報Ihに基づいて、サービス識別設定器312は、サービス親子情報Escを生成する。例えば、サービス親子情報Ihに基づいて、サービス親子指定情報ISH(S1-1)が生成される。サービス親子情報Esc(S1-1)は、このサービスが親サービスS1の子サービスであることを定義する。

【0116】

付随情報ISが実行形式DEに付与されて識別化実行形式DEiが生成され、サービス親子指定情報ISHが実行形式DEに付与されて親子識別化実行形式DEcが生成される。つまり、サービス識別設定器312からは、識別化実行形式DEiと親子識別化実行形式DEcが混在して出力される。そして、これらの識別化実行形式DEiと親子識別化実行形式DEcは、送出器113およびデリバリシステム120を介して端末330に入力される。そして、端末330のダウンローダ331によって、識別化実行形式DEiと親子識別化実行形式DEcは識別化実行形式格納器332に出力される。さらに、ダウンローダ331は、識別化実行形式DEiから付随情報ISを抽出し、親子識別化実行形式DEcからサービス親子指定情報ISHを、それぞれをサービス依存リソース管理器136に出力する。

【0117】

図12に、サービス識別設定器312から出力された識別化実行形式DEiと親子識別化実行形式DEcが識別化実行形式格納器332に格納される様子を示す。同図に示す例においては、図2に示した第1の実施の形態におけるのと同様に、サービスS1、サービスS2乃至サービスSn(nは任意の自然数)に対応する識別化実行形式DEi(S1)、DEi(S2)、およびDEi(Sn)が

例示されている。ただし、本図においては、サービス S 3 に対応する識別化実行形式 DE i (S 3) の位置に、初めての親子識別化実行形式 DE c (1) が表示されている。親子識別化実行形式 DE c (1) は、実行形式 DE に、それが子サービスであることを示すサービス親子指定情報 IS h (1) が付与されている。

【0118】

親子識別化実行形式 DE c (1) は、それが親サービス S p 1 に対応するサービス識別情報 E s (S c 1) と副次情報 α c 1 から成るサービス親子指定情報 IS h (1) と実行形式 DE を含む。具体的には、親子サービス識別情報 E s c が実行形式 DE を親サービス S p 1 に対する子サービス S c 1 として対応付けている。このようにサービス識別情報 E s (S p 1) の「S p 1」が子サービス 1 を規定する点を除けば、サービス親子指定情報 IS h も基本的には付随情報 IS と同じものである。

【0119】

つまり、子サービス S c は、上述のように、サービス S n の 1 つであるので、親子識別化実行形式 DE c は、識別化実行形式 DE i (S c) と表すこともできる。よって、サービス親子指定情報 IS h も付随情報 IS と総称してもよいが、本明細書においては、本実施の形態における特徴をわかりやすくするために、サービス親子指定情報 IS h を付随情報 IS と区別して説明する。さらに、親サービス S p に対する子サービス S c を規定するサービス識別情報 E s を親子サービス識別情報 E s c と称して、実行形式 DE に対応するサービス識別情報 E s と区別して説明する。

【0120】

識別化実行形式格納器 332 の動作は、上述の第 1 の実施の形態にかかる 132 と基本的に同じである。ただし、識別化実行形式格納器 332 は、識別化実行形式 DE i と親子識別化実行形式 DE c とを格納し、要求に応じて、識別化実行形式 DE i あるいは親子識別化実行形式 DE c をエグゼキュータ 333 に出力する。

【0121】

エグゼキュータ 333 は、識別化実行形式 DE i からサービス識別情報 E s を

抽出し、親子識別化実行形式DE cから親子サービス識別情報E s cを抽出して、それぞれを親子判定器334に出力する。

【0122】

親子判定器334は、リソースセクタ134から入力されるAPI呼出C a p iと、エグゼキュタ333から入力されるサービス識別情報E sおよび親子サービス識別情報E s cに基づいて、実行中の実行形式DEがAPI呼出C a p iの処理対象のサービス依存リソースのサービスの先祖（親、または親の親、または親の親の親．．．）である場合にのみ、API呼出C a p iをサービス依存リソース管理者136に出力する。

【0123】

つまり、親子判定器334は、実行中の実行形式DEの親子サービス識別情報E s cをエグゼキュタ333から得るとともに、エグゼキュタ333から発行されるAPI呼出C a p iをリソースセクタ134を介して得る。さらに、ダウンローダ331からサービス間の親子関係を判定するための情報であるサービス親子指定情報I S hを得る。そして、親子サービス識別情報E s cが示すサービスが、サービス親子指定情報I S hの示すサービスの先祖であるかを判断する。そして、先祖であると判断した場合には、サービス依存リソース管理者136に対してサービス依存APIの呼び出し要求（API呼出C a p i）を伝え、先祖でないと判断した場合には伝えない。

【0124】

次に、図13に示すフローチャートを参照して、端末330におけるサービス実行ルーチンの動作について説明する。具体的には、端末330において、エグゼキュタ333が呼び出すAPIに対する実行形式DEおよび親子識別化実行形式DE cに含まれる実行形式DEを実行させるために、API呼出C a p iを発行した時点で、本ルーチンにおける処理が開始される。

【0125】

なお、図13に示すフローチャートは、上述の図5に示すフローチャートにおいて、「実行中の実行形式DEのサービス識別情報E sを実行エンジンから取得」ステップS514が「実行中の実行形式DEのサービス識別情報E sまたは親

子サービス識別情報 E s c を取得」ステップ S 1 3 1 4 に交換され、「実行中のサービスに対する操作かを判定」ステップ S 5 1 6 が「実行中のメタサービスに対する操作かを判定」ステップ S 1 3 1 6 に交換されている点を除いては同様に構成されている。以下、本実施の形態に固有の動作に重点をおいて説明する。

【0126】

よって、ステップ S 5 1 2 において、リソースセクタ 1 3 4 によって、エグゼキュタ 3 3 3 から発行された A P I 呼出 C a p i に基づいて、呼び出された A P I がサービス依存 A P I であると判断されて、処理はステップ S 1 3 0 4 に進む。

【0127】

ステップ S 1 3 1 4 において、親子判定器 3 3 4 によって、エグゼキュタ 3 3 3 から現在実行中の実行形式 D E に対するサービス識別情報 E s あるいは親子サービス識別情報 E s c が読み出される。そして、処理は次のステップ S 1 3 0 6 に進む。

【0128】

ステップ S 1 3 1 6 において、親子判定器 3 3 4 によって、サービス依存 A P I 呼出 C a p i の処理対象のサービス依存リソースに対応するサービスに対して、実行中の実行形式 D E の対応するサービスが先祖のサービスであるか否かが判断される。N o の場合、上述のエラー処理ステップ S 5 2 1 を経て、本ルーチンが終了される。一方、実行中の実行形式 D E が実行中のサービスが先祖である場合は、Y e s と判断されて、処理は上述の「サービス依存リソースに対して処理」ステップ S 5 1 8 を経て本ルーチンが終了される。

【0129】

なお、ステップ S 5 1 8 においては、サービス依存リソース管理者 1 3 6 によって、サービス依存 A P I 呼出 C a p i に対応するサービス依存リソースに対する処理が実行される。すなわち、現在実行している実行形式 D E のサービスの子孫（子、または子の子、または子の子の子、．．．）のサービス依存リソースに対して処理が可能になるように管理される。

【0130】

(第4の実施の形態)

次に、図14および図15、図16、および図17を参照して、本発明の第4の実施の形態にかかるサービス安全拡張プラットフォームについて説明する。図8に示すように、本実施の形態の形態にかかるサービス安全拡張プラットフォームSEP4は、図1に示すサービス安全拡張プラットフォームSEP1におけるサービス提供源110がサービス提供源410に変更され、端末130が端末430に変更されている。

【0131】

サービス提供源410は、サービス提供源110における実行形式提供器111がコンテンツ提供器411に交換され、サービス識別設定器112が制御コンテンツ指定器412に交換されている。端末430は、端末130におけるダウンロード131がコンテンツダウンロード431に交換され、識別化実行形式格納器132がコンテンツ格納器432に交換され、エグゼキュータ133がエグゼキュータ433に交換されると共に、リソースセクタ134とサービス依存リソース管理者136との間に制御コンテンツ判定器434が新たに設けられている。

【0132】

コンテンツ提供器411は、サービスを実現する実行形式DEと実行形式DEが解釈しユーザに提示する目的で製作されたデータであるコンテンツDCを格納し、必要に応じて出力する。実行形式DEもコンテンツDCの満たすべき要件を備えている。例えば、共に、実行形式DEもコンテンツDCも共に、逐次動作手順をプログラミングするスクリプト記述を含めることが可能なマークアップ言語で記述する。スクリプトの言語としてJava(R)系のJavaScriptやECMAScriptなどの言語でよい。また、マークアップ言語としては、HTML言語やBML言語でよい。

【0133】

制御コンテンツ指定器412は、コンテンツ提供器411が出力するコンテンツDCのうち、制御コンテンツに対して制御コンテンツであることを指定する付随情報ISCを付与して制御コンテンツDCcとして出力する。制御コンテンツ

DC c の定義は、サービスに対応付けられた実行形式DEとして製作されたコンテンツDCである。

【0134】

コンテンツダウンローダ431は、デリバリシステム120から伝送されてくる、コンテンツDCと制御コンテンツDC c とを受信し、コンテンツ格納器432に受信した情報を書き込む。

【0135】

デリバリシステム120からは実行形式DEもコンテンツDCの一種類として伝送されるため、コンテンツダウンローダ431は、コンテンツDCを格納する機能を有していればよい。

【0136】

コンテンツダウンローダ431は、サービス依存リソース管理者136がサービス依存リソースとして保持する、サービスのコンテンツを自動ダウンロードすべきかを表す情報に基づきコンテンツの格納を制御する。

すなわち、コンテンツダウンローダ431は、注目するサービスに対し、自動ダウンロードすべき場合には、サービスに関するコンテンツDCを制御コンテンツDC c も含め全て格納する。一方、自動ダウンロードすべきでない場合には、制御コンテンツDC c のみを格納する。

【0137】

コンテンツ格納器432は、コンテンツDCを格納する。第1の実施の形態の識別化実行形式格納器132が格納する情報が実行形式DEであるのに対し、コンテンツ格納器432はコンテンツDCを格納する。なお、コンテンツDCには制御コンテンツDC c が含まれる。

【0138】

図15に、制御コンテンツ指定器412から出力されたコンテンツDCと付随情報ISCがコンテンツ格納器432に格納される様子を示す。同図に示す例においては、サービスS1に対して、制御コンテンツDC c であるC(S1, DE)、コンテンツDCであるC(S1, 1)、C(S1, 2)、およびC(S1, 3)がコンテンツ格納器432に保持されている。制御コンテンツDC c である

C (S1, E) をエグゼキュータ 433 で実行させることで、コンテンツ DC である C (S1, 1)、C (S1, 2)、および C (S1, 3) を読み込んでユーザに提示する。すなわち制御コンテンツ DC c である C (S1, DE) は、コンテンツ DC である C (S1, 1)、C (S1, 2)、C (S1, 3) に対するいわゆるブラウザと同等の動作を行う。

【0139】

制御コンテンツ判定器 434 は、エグゼキュータ 433 の出力から、実行しているコンテンツ DC が制御コンテンツ DC c であるかを判定する。そして、制御コンテンツ DC c である場合のみに、リソースセクタ 134 からのサービス依存 API の要求をサービス依存リソース管理者 136 に伝える。

【0140】

図 16 に、本実施の形態において、サービス依存リソース管理者 136 が格納するサービス依存リソーステーブル Trs4 の一例である。サービス依存リソーステーブル Trs4 は、図 3 に示した、第 1 の実施の形態にかかるサービス依存リソース管理テーブル Trs と類似しているが、列 C2 には、コンテンツダウンローダ 431 がコンテンツ DC を自動ダウンロードすべきかについての情報がサービス毎に保持されている。

【0141】

次に、図 17 に示すフローチャートを参照して、端末 430 における制御コンテンツ DC c の実行ルーチンの動作について説明する。具体的には、端末 430 において、エグゼキュータ 433 が呼び出す API に対する制御コンテンツ DC c を実行するために、API 呼出 Cap i を発行した時点で本ルーチンにおける処理が開始される。

【0142】

なお、図 17 に示すフローチャートは、上述の図 5 に示すフローチャートにおいて、実行中の実行形式 DE のサービス識別情報 Es を実行エンジンから取得」ステップ S514 が「実行中のコンテンツ DC のサービス識別情報 Es を取得」ステップ S1714 に交換され、「実行中のサービスに対する操作かを判定」ステップ S516 が「実行中のサービスに対する制御コンテンツ操作かを判定」ス

テップS1716に交換されている点を除いては同様に構成されている。以下、本実施の形態に固有の動作に重点をおいて説明する。

【0143】

ステップS512において、リソースセクタ134によって、エグゼキュータ333から発行されたAPI呼出Capiに基づいて、呼び出されたAPIがサービス依存APIであると判断されて、処理はステップS1704に進む。

【0144】

ステップS1714において、制御コンテンツ判定器434によって、エグゼキュータ433から現在実行中のコンテンツDCに対するサービス識別情報Esが読み出される。そして、処理は次のステップS1716に進む。

【0145】

ステップS1716において、制御コンテンツ判定器434によって、サービス依存API呼出Capiの処理対象のサービス依存リソースに対応するサービスに対して、実行中のコンテンツDCがコンテンツDCの対応するサービスに対応する制御コンテンツであるか否かが判断される。Noの場合、上述のエラー処理ステップS521を経て、本ルーチンが終了される。一方、実行中のコンテンツDCが実行中のサービスが先祖である場合は、Yesと判断されて、処理は上述の「サービス依存リソースに対して処理」ステップS518を経て本ルーチンが終了される。

【0146】

なお、ステップS518においては、サービス依存リソース管理者136によって、サービス依存API呼出Capiに対応するサービス依存リソースに対する処理が実行される。

【0147】

本明細書で開示したサービス安全拡張プラットフォームによれば、サービスを実現する実行形式DEによる不正アクセスの排除を、実行形式DEに不正アクセスを引き起こす処理が含まれることを否定する必要無く達成するという、新次元の安全性を実現することができる。

【0148】

詳述すれば、他のサービスの状態変更やデータの破壊や、プラットフォーム自体のシステムダウンなど、サービスの正常な享受に悪影響を及ぼす挙動を引き起こす処理である不正アクセスの排除を、実行形式DEに不正アクセスを引き起こす処理が含まれることを実行形式DEの配布元の確からしさから推測するまでもなく達成する。

【0149】

さらには上記の性質と同時に、実行形式DEの変更や追加によってサービスの機能／諸元の変更や新規サービスの追加といったサービスの拡張も達成する機能を実現することができる。

これにより、従来のサービスの拡張可能なプラットフォームに対して従来になり新次元の安全性と操作性とを同時に実現することができる。

【図面の簡単な説明】

【図1】

本発明の第1の実施の形態にかかる、サービス安全拡張プラットフォームの構成を模式的に示すブロック図である。

【図2】

図1に示したサービス安全拡張プラットフォームの識別化実行形式格納器に格納される識別化実行形式を示す説明図である。

【図3】

図1に示したサービス安全拡張プラットフォームのサービス依存リソース管理者で管理されるサービス依存リソース管理テーブルの説明図である。

【図4】

図1に示したサービス安全拡張プラットフォームのサービス依存リソース管理者による新サービスの登録管理動作を示すフローチャートである。

【図5】

図1に示したサービス安全拡張プラットフォームの端末によるサービス実行動作を示すフローチャートである。

【図6】

図1に示したサービス安全拡張プラットフォームの端末のソフトウェア階層を

示す説明図である。

【図 7】

図 1 に示したサービス安全拡張プラットフォームによってユーザに提示される画面例を示す説明図である。

【図 8】

本発明の第 2 の実施の形態にかかるサービス安全拡張プラットフォームの構成を模式的に示すブロック図である。

【図 9】

図 8 に示したサービス安全拡張プラットフォームの識別化実行形式格納器に格納される識別化実行形式およびメタサービス実行形式を示す説明図である。

【図 10】

図 8 に示したサービス安全拡張プラットフォームの端末によるサービス実行動作を示すフローチャートである。

【図 11】

本発明の第 3 の実施の形態にかかるサービス安全拡張プラットフォームの構成を模式的に示すブロック図である。

【図 12】

図 11 に示したサービス安全拡張プラットフォームの識別化実行形式格納器に格納される識別化実行形式および親子識別化実行形式を示す説明図である。

【図 13】

図 11 に示したサービス安全拡張プラットフォームの端末によるサービス実行動作を示すフローチャートである。

【図 14】

本発明の第 4 の実施の形態にかかるサービス安全拡張プラットフォームの構成を模式的に示すブロック図である。

【図 15】

図 14 に示したサービス安全拡張プラットフォームのコンテンツ格納器に格納される制御コンテンツおよびコンテンツを示す説明図である。

【図 16】

図 1 4 に示したサービス安全拡張プラットフォームのサービス依存リソース管理
器で管理されるサービス依存リソース管理テーブルの説明図である。

【図 1 7】

図 1 4 に示したサービス安全拡張プラットフォームの端末による制御コンテン
ツ実行動作を表すフローチャートである。

【図 1 8】

従来の技術における、従来のサービスプラットフォームの構成を模式的示すブ
ロック図である。

【図 1 9】

図 1 8 に示したサービスプラットフォームの識別化実行形式格納器に格納され
る実行ファイルを示す説明図である。

【図 2 0】

図 1 8 に示したサービスプラットフォームの端末におけるのソフトウェア階層
を示す説明図である。

【符号の説明】

SEP 1 ～ SEP 4 サービス安全拡張プラットフォーム

110、210、310、410 サービス提供源

111 実行形式提供器

113 送出器

112 サービス識別設定器

120 デリバリシステム

130、230、330、430 端末

131、231、331、 ダウンローダ

132、332、 識別化実行形式格納器

134 リソースセ렉タ

135 一般リソース管理器

136 サービス依存リソース管理器

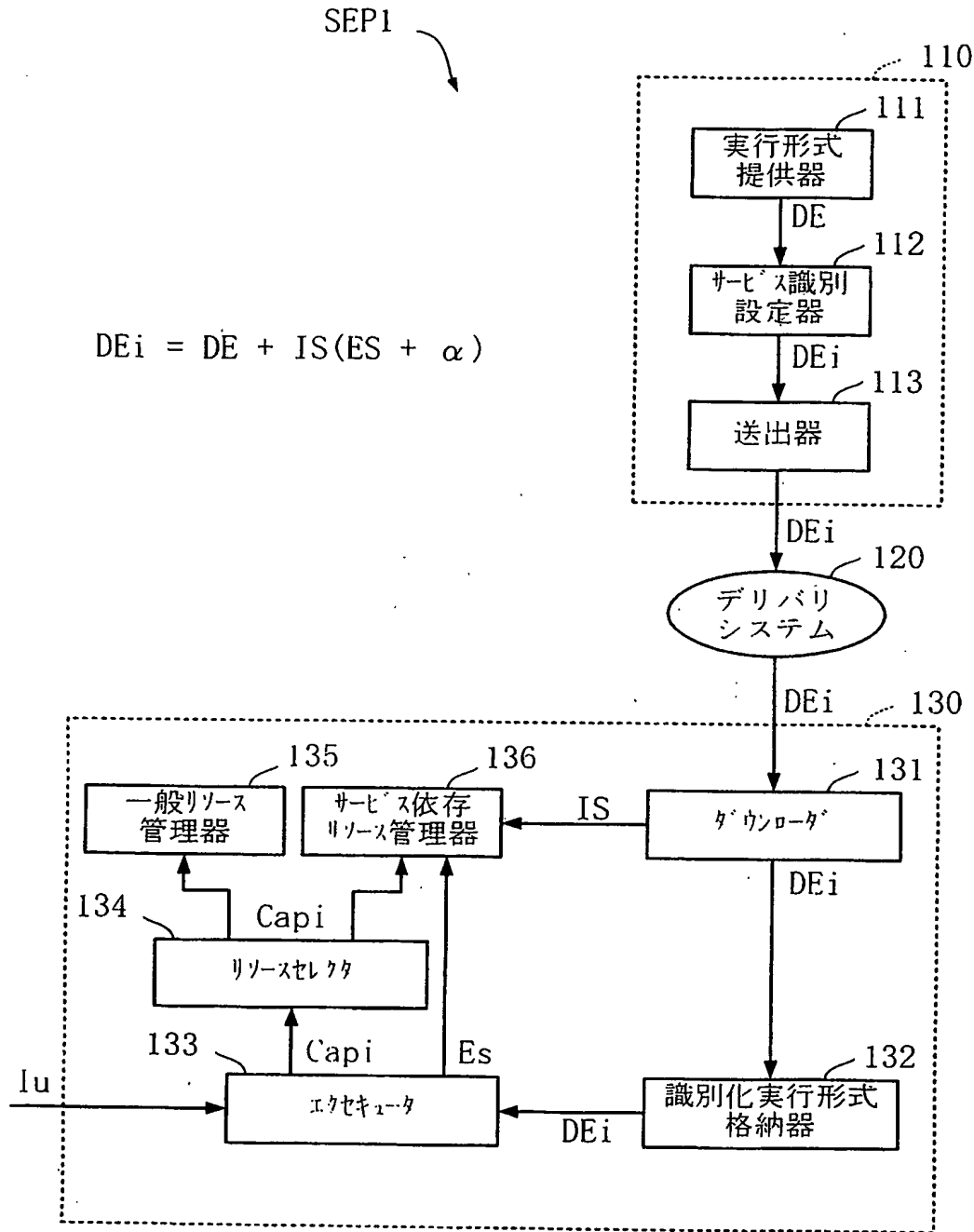
212 メタサービス指定器

234 メタサービス判定器

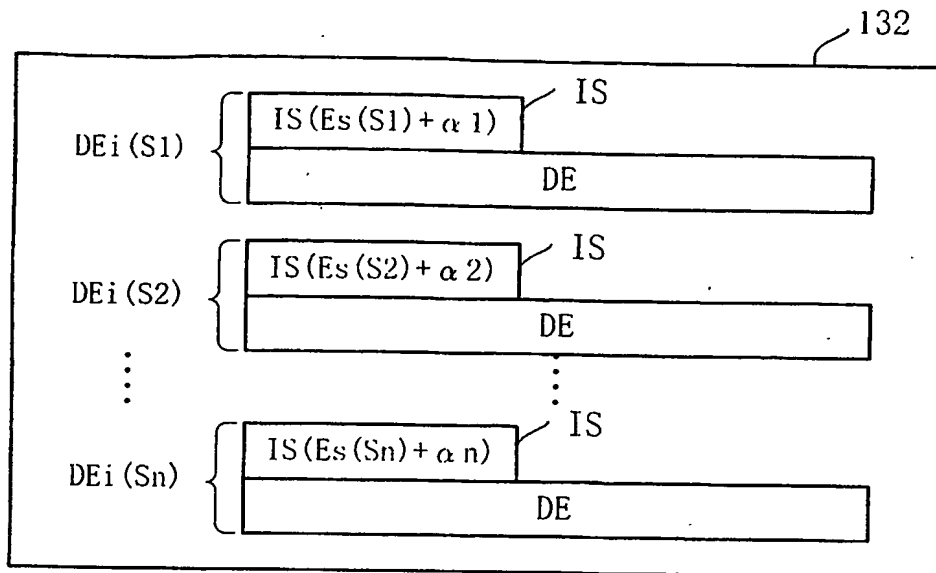
- 3 1 2 サービス識別設定器
- 3 3 4 親子判定器
- 4 1 1 コンテンツ提供器
- 4 1 2 制御コンテンツ指定器
- 4 3 1 コンテンツダウンローダ
- 4 3 2 コンテンツ格納器
- 4 3 4 制御コンテンツ判定定器

【書類名】 図面

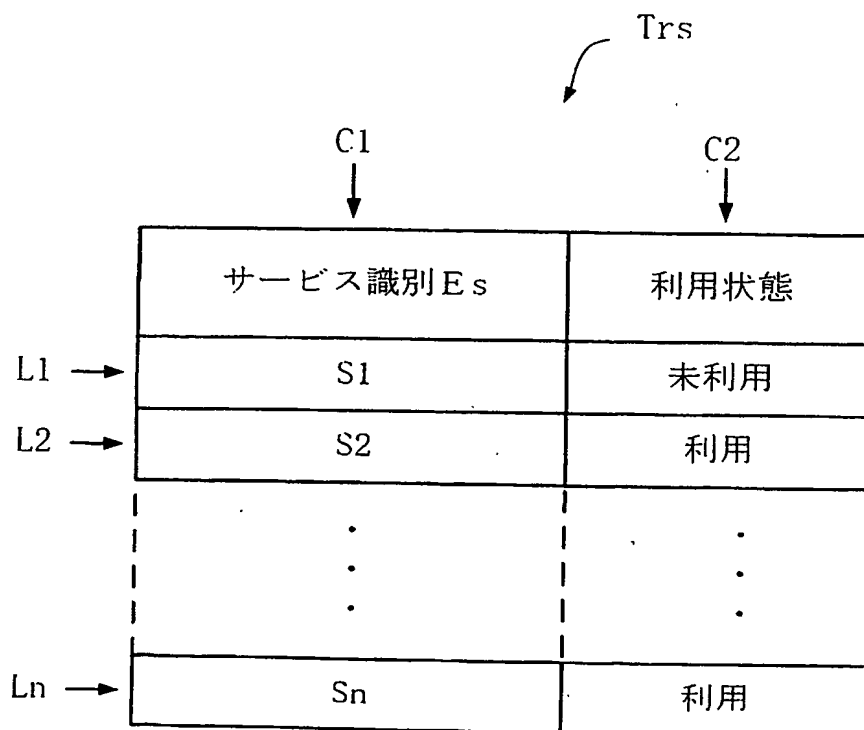
【図 1】



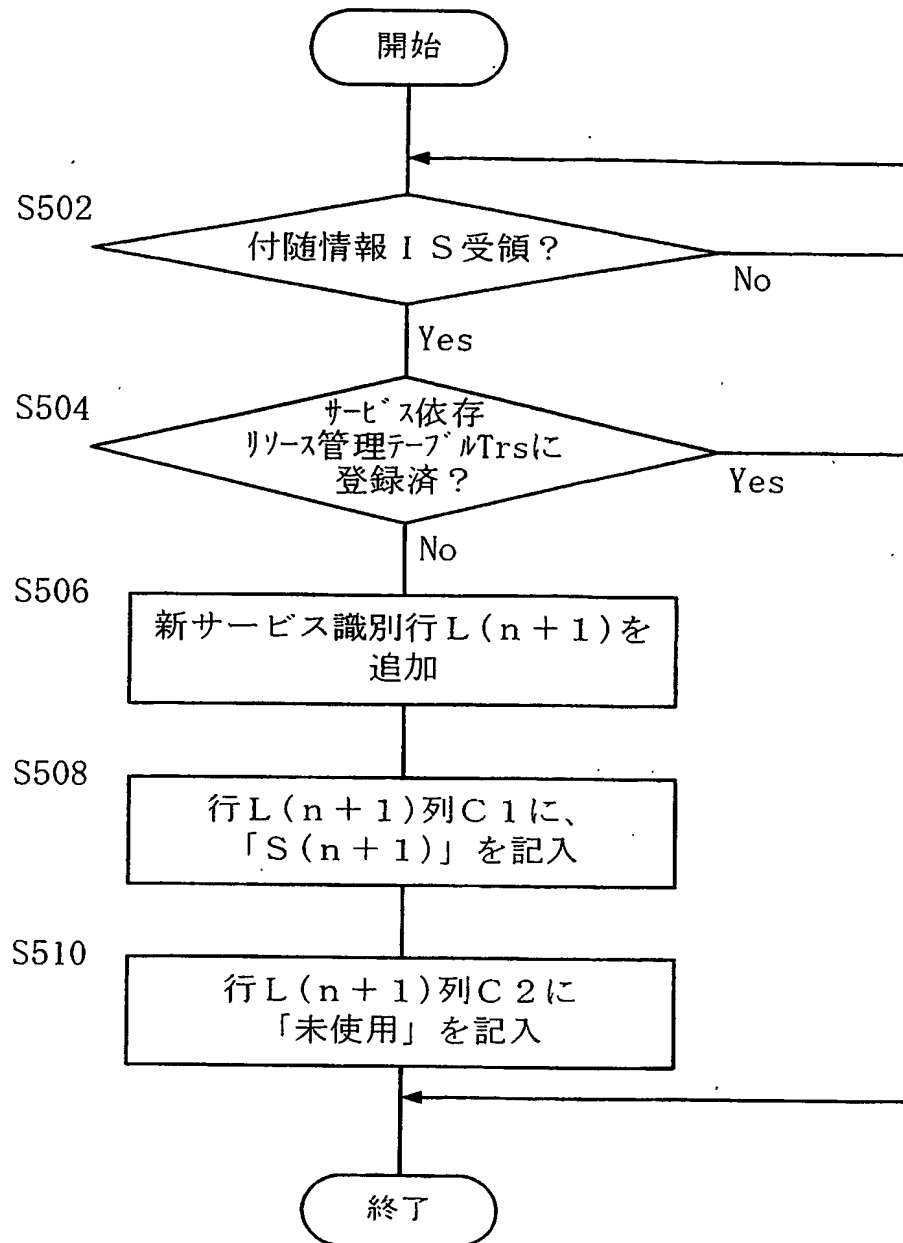
【図 2】



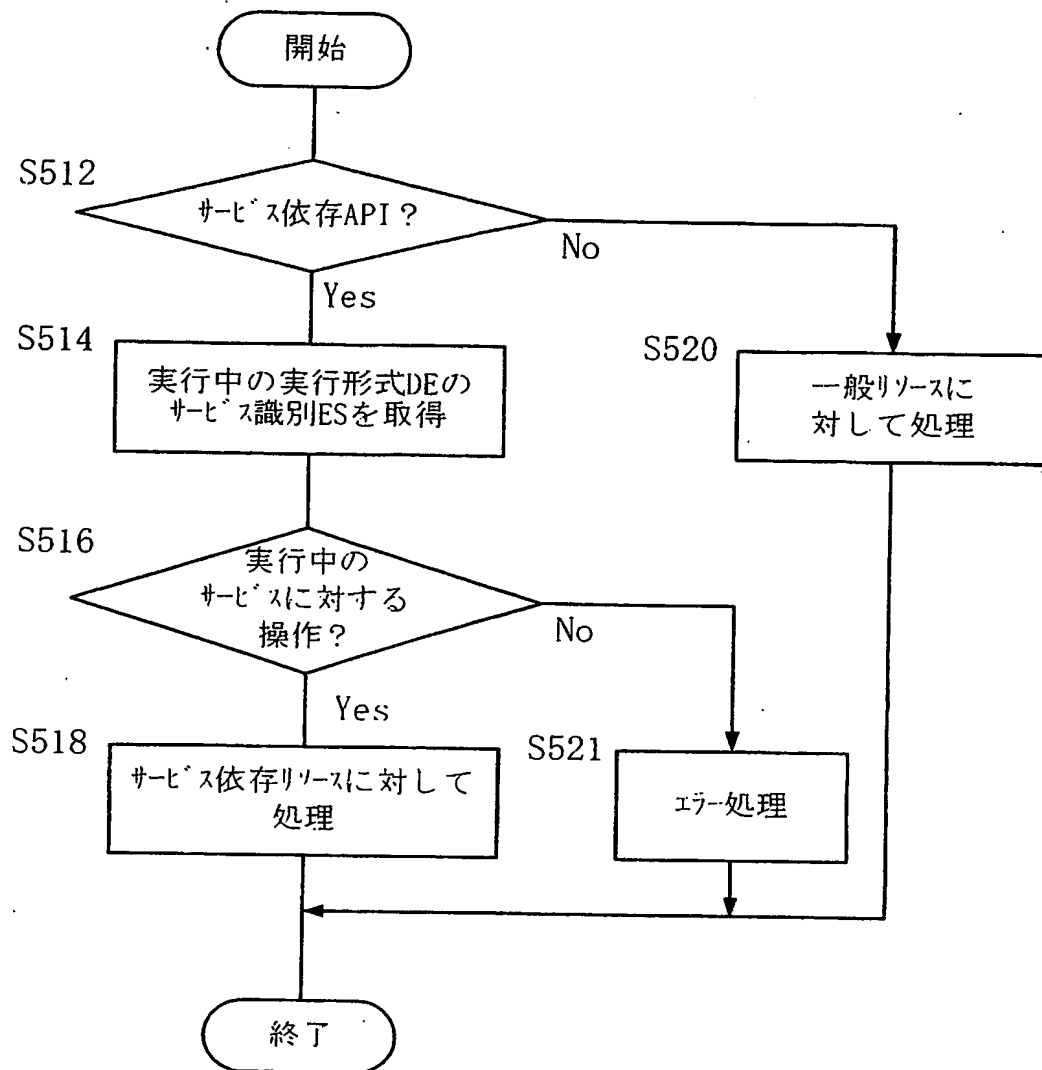
【図 3】



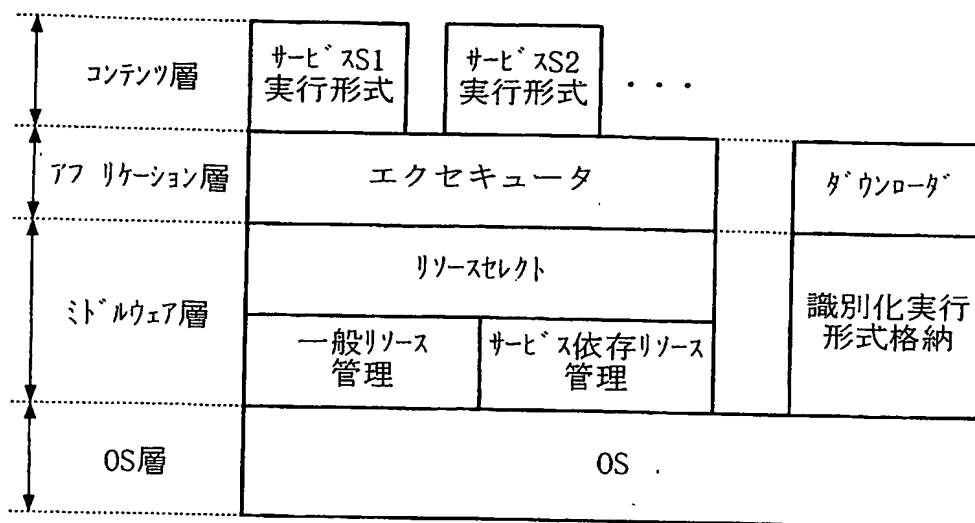
【図 4】



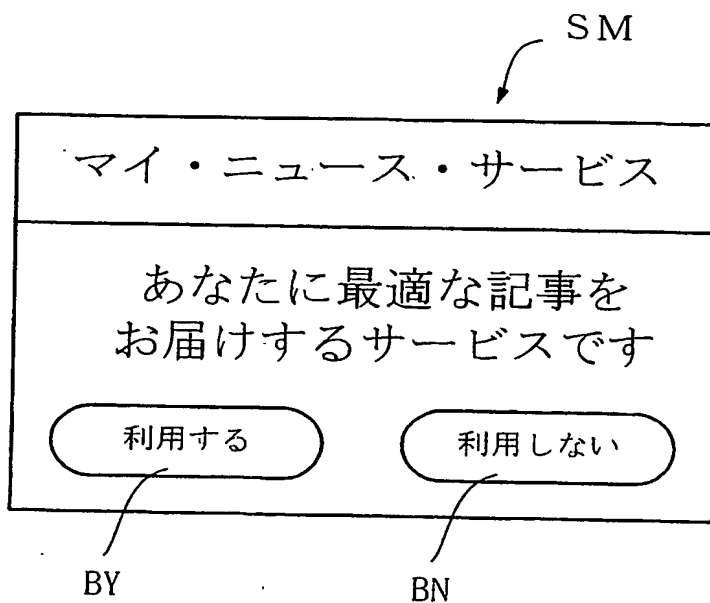
【図5】



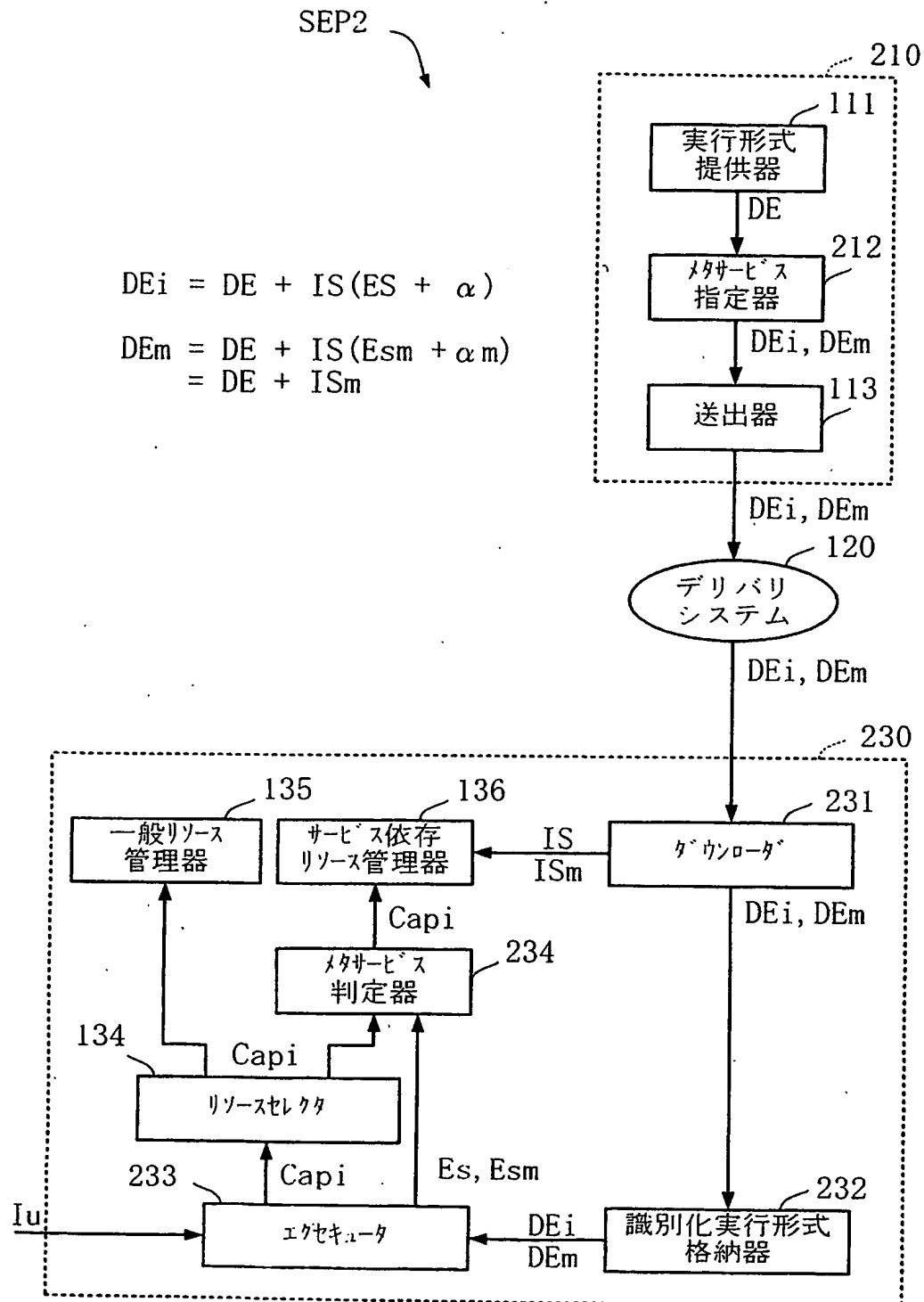
【図6】



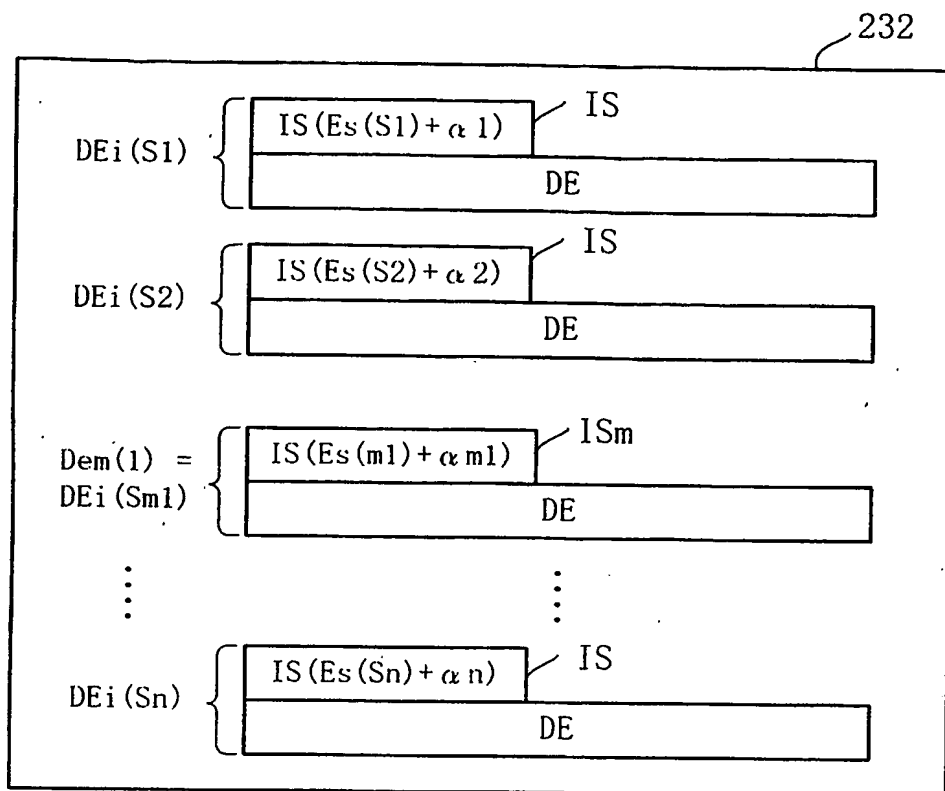
【図7】



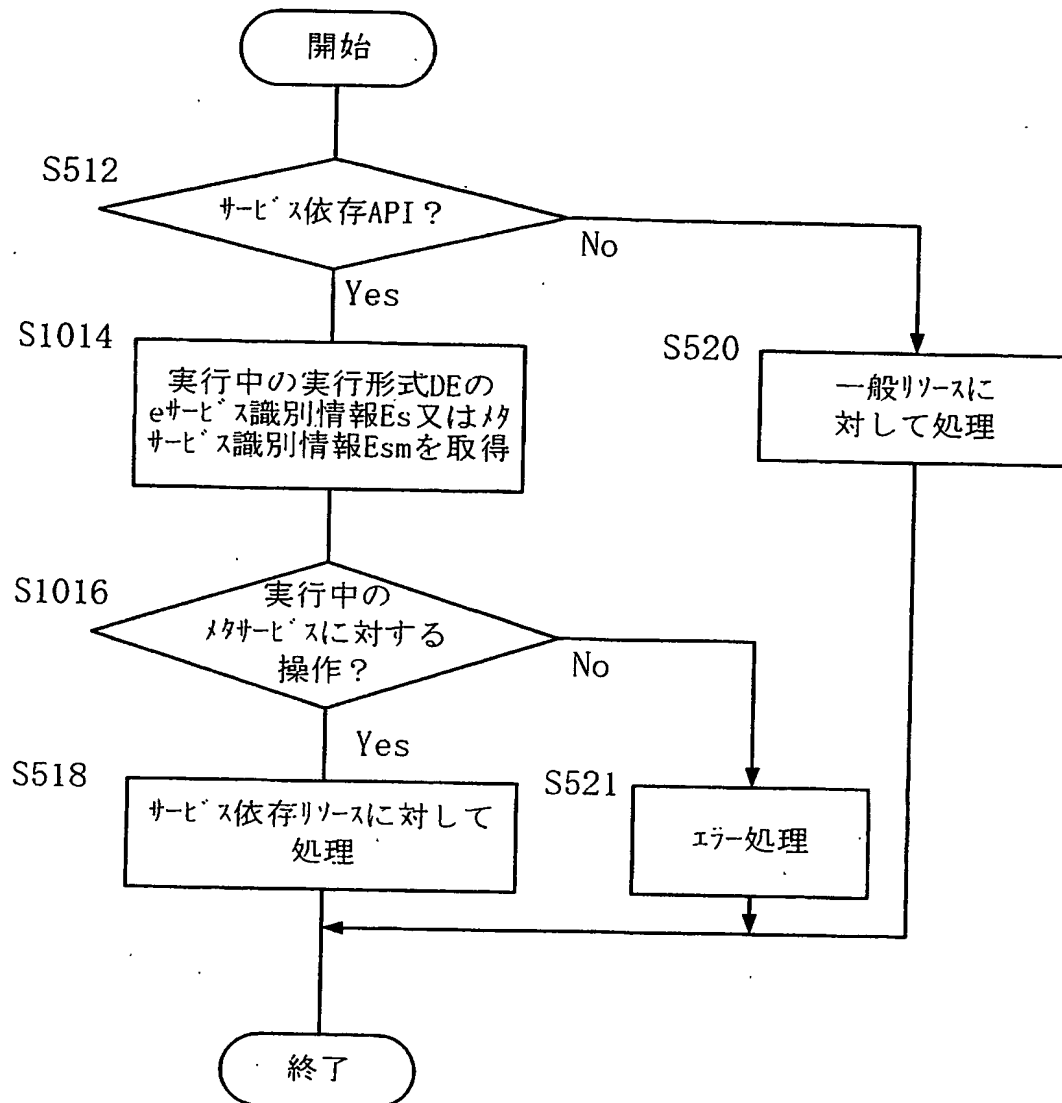
【図 8】



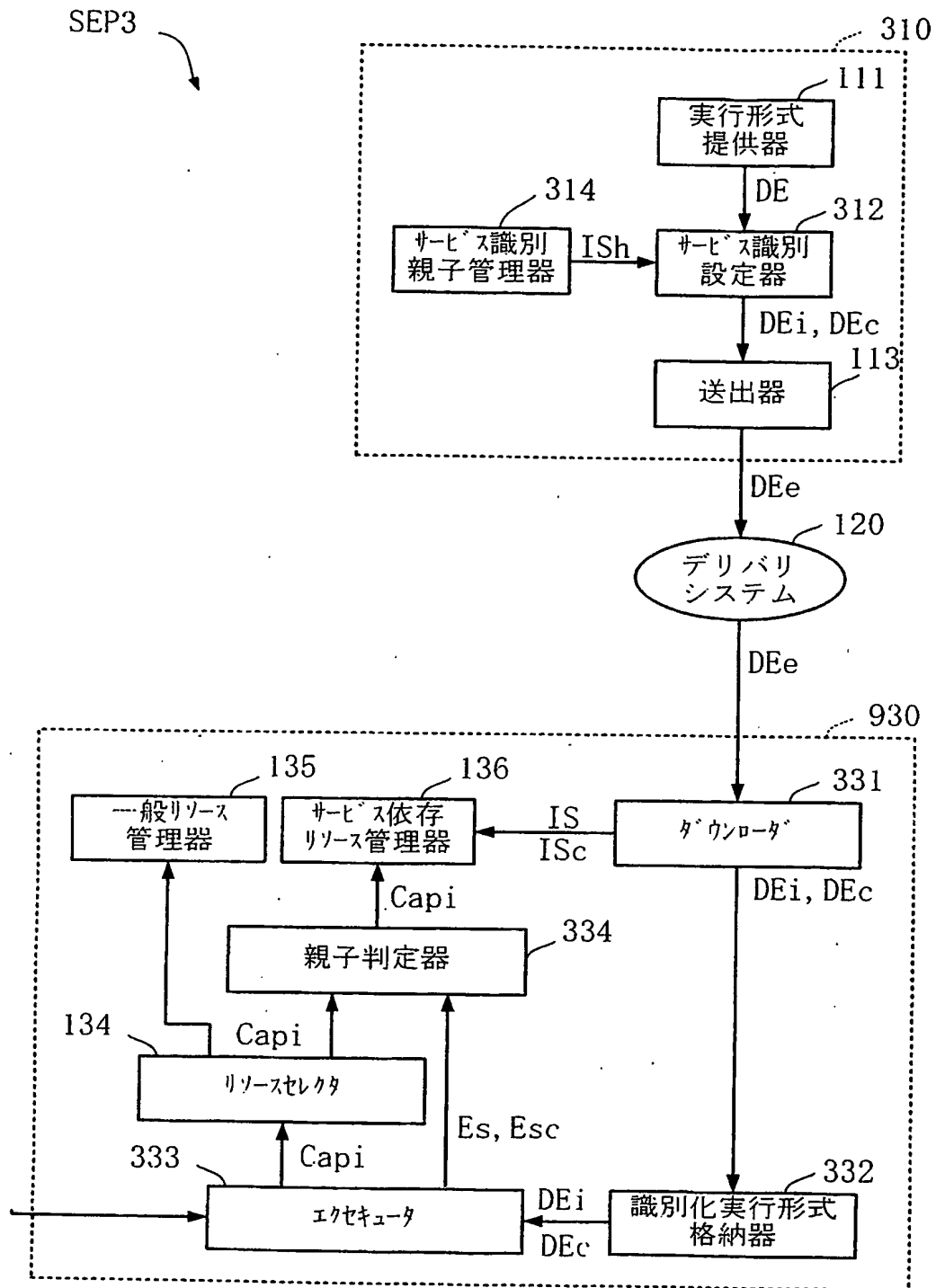
【図 9】



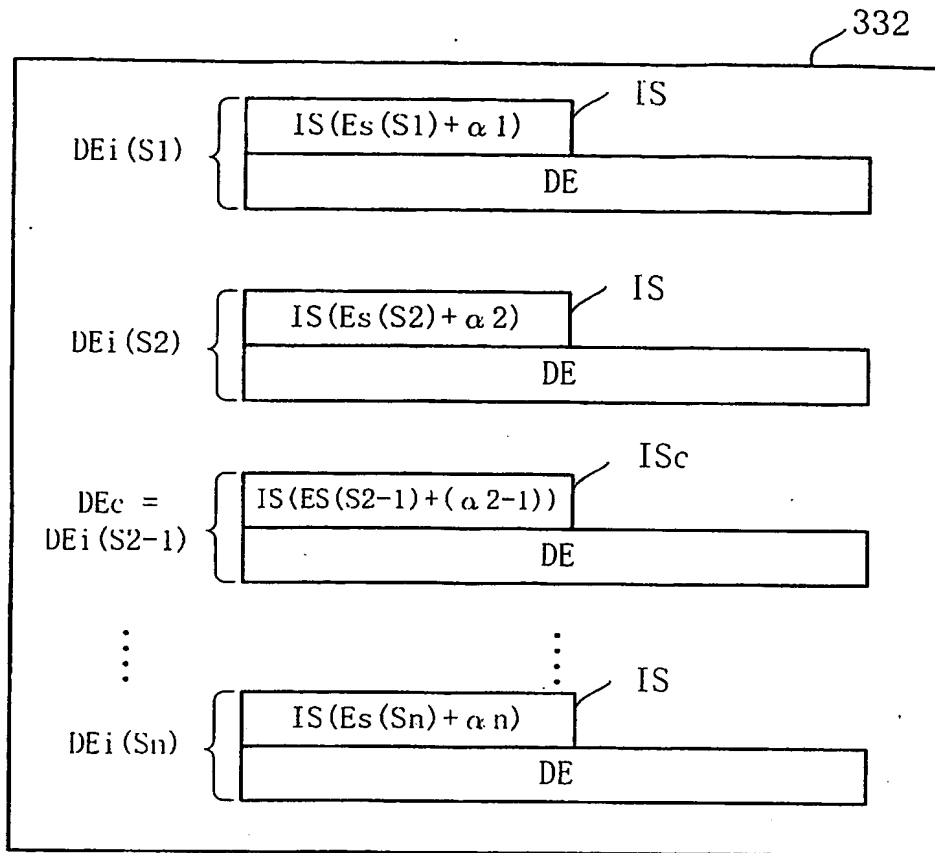
【図10】



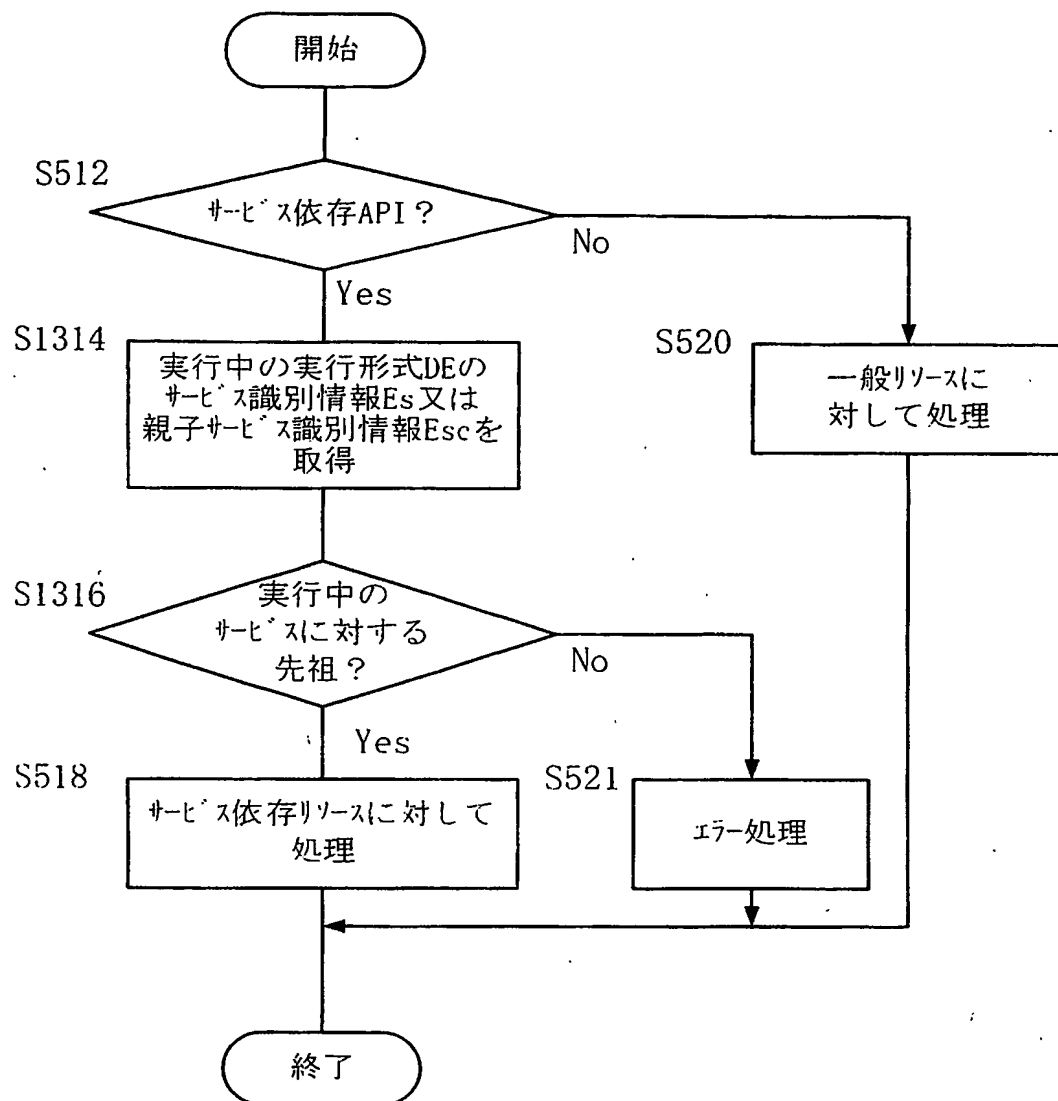
【図 11】



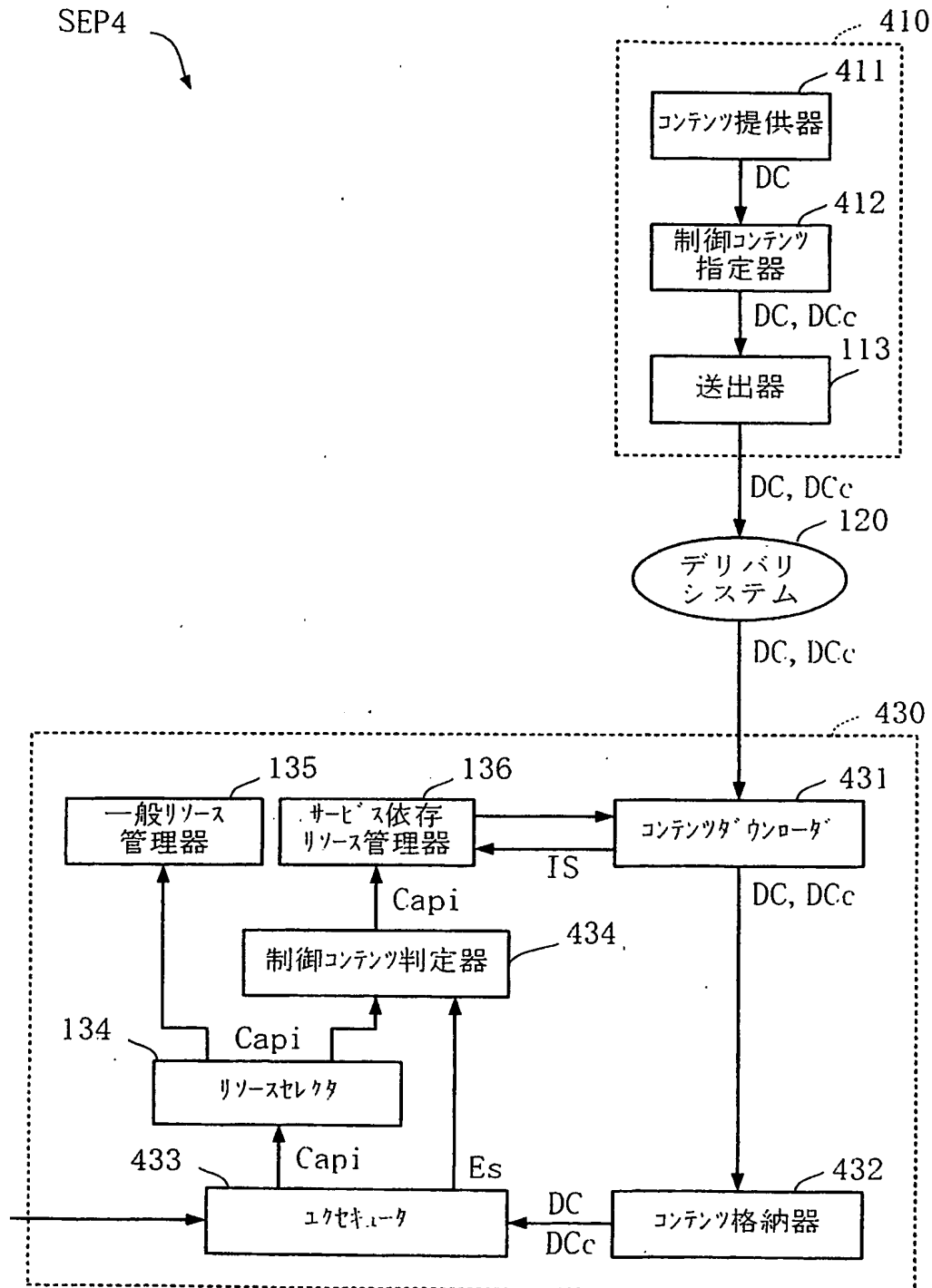
【図 1 2】



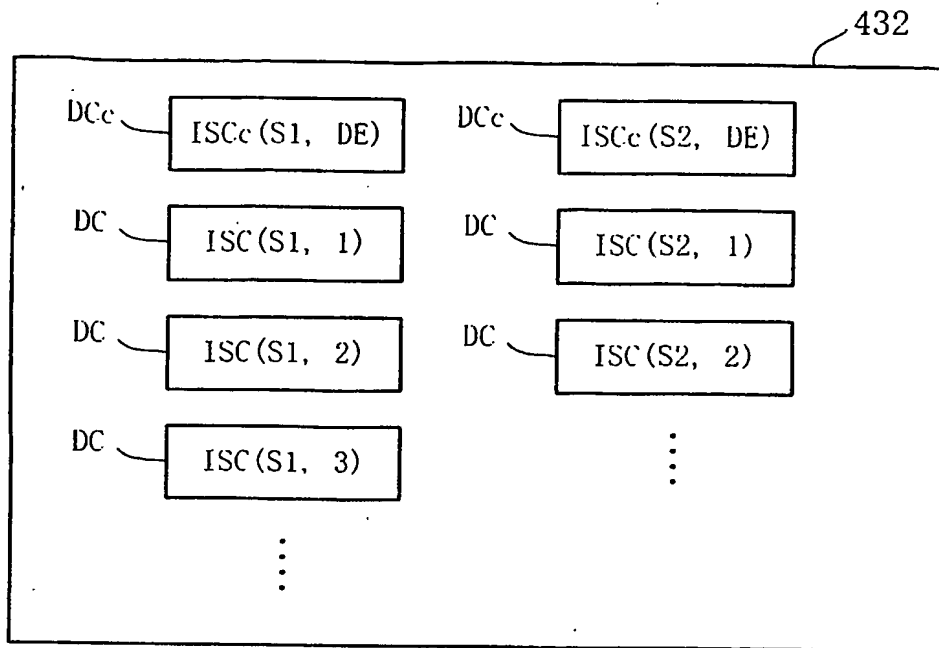
【図 1 3】



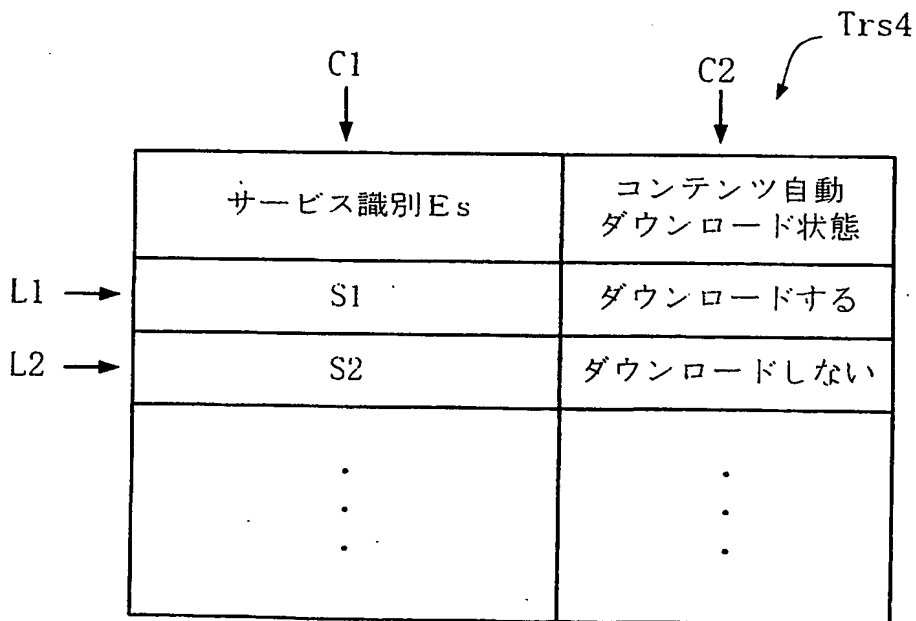
【図14】



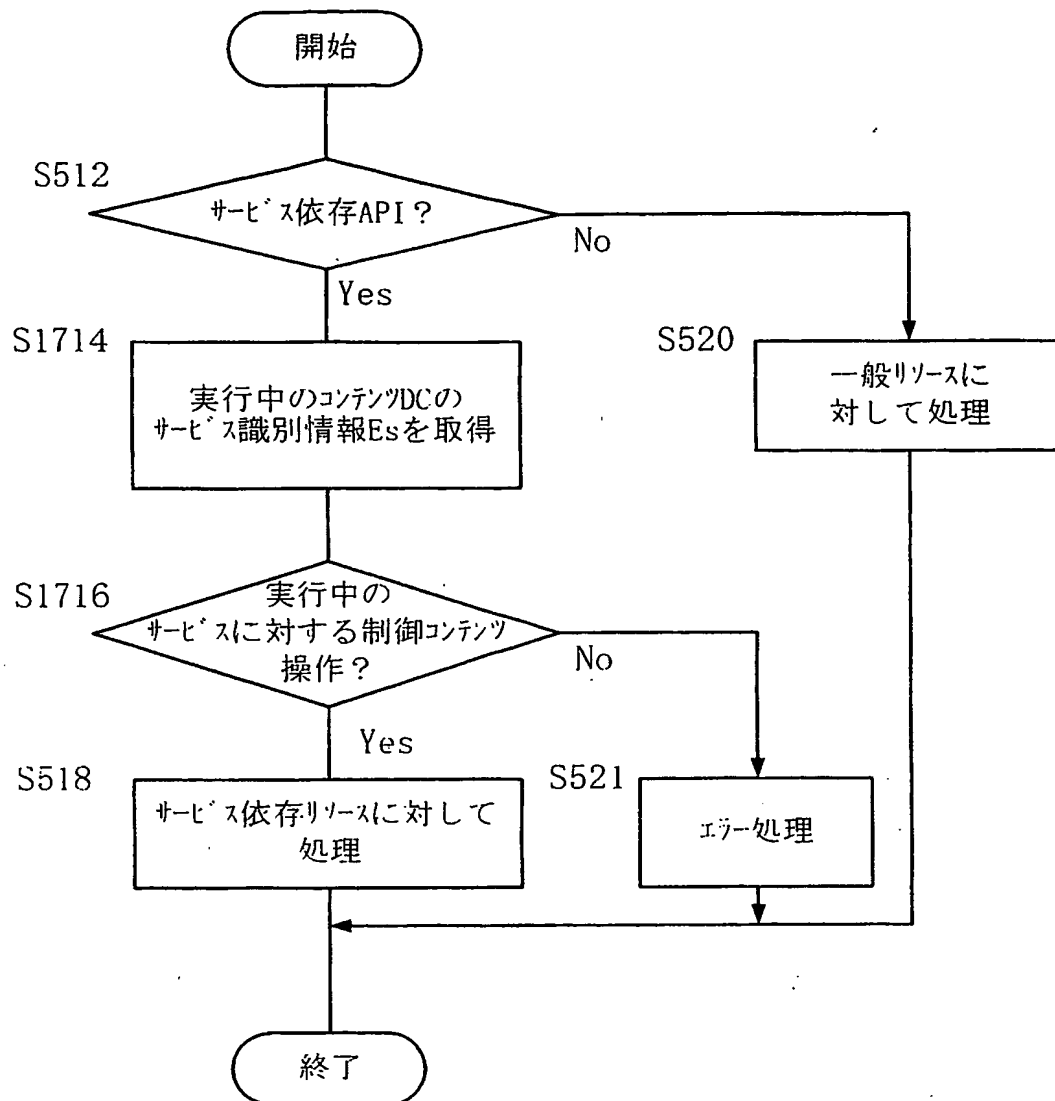
【図 15】



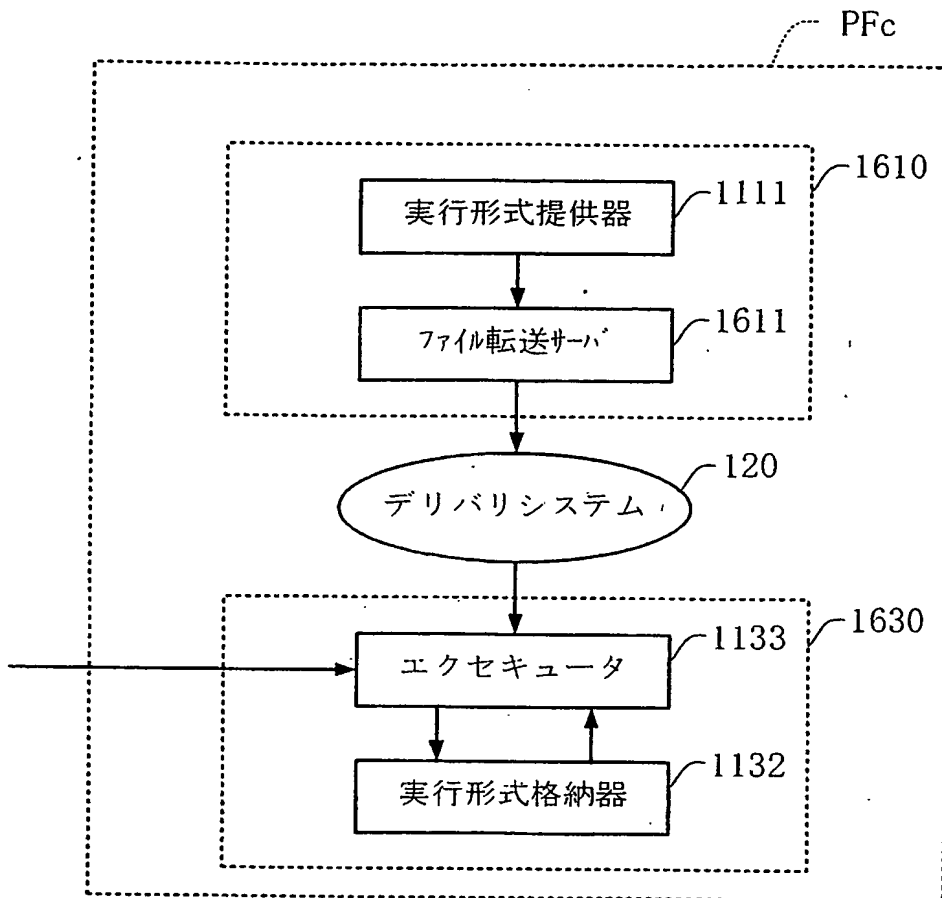
【図 16】



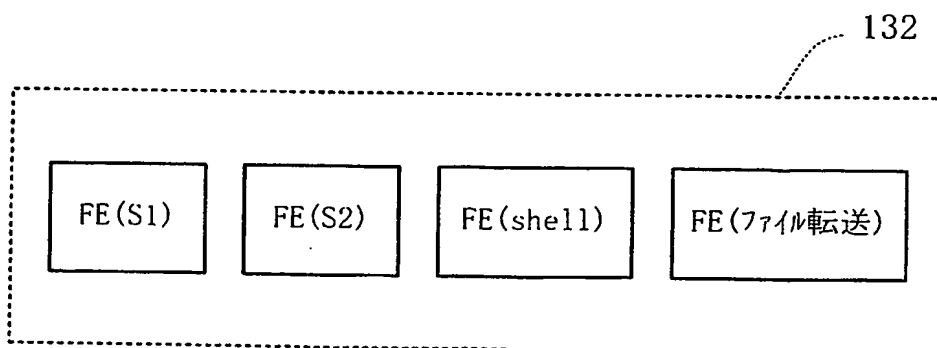
【図 17】



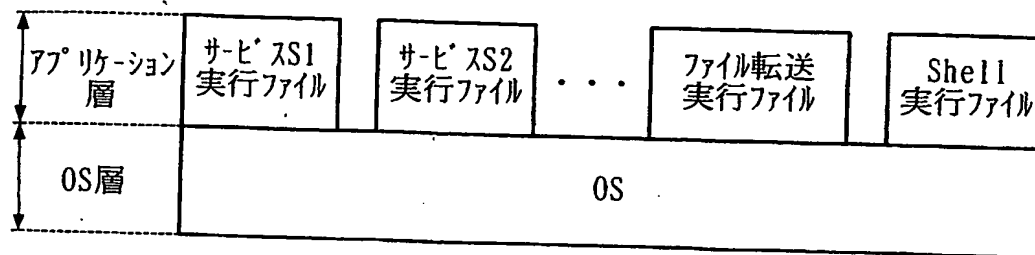
【図18】



【図19】



【図 20】



【書類名】 要約書

【要約】

【課題】 実行形式に不正アクセスを引き起こす処理が含まれることを否定する必要無く、デジタルコンテンツの提供サービスを実現する実行形式による不正アクセスの排除できる、サービスの拡張可能な全く新しいサービス安全拡張プラットフォームを提供することを目的とする。

【解決手段】 サービス（S）と実行形式（DE）とが対応付けられており、前記実行形式（DE）の変更や追加によって前記サービス（S）の拡張が達成されるサービス安全拡張プラットフォーム（SEP）は、サービスの拡張を行うサービス依存APIを備え、かつ実行形式（DE）からのサービスの拡張はサービス依存APIの呼び出しによってのみ行われる。

【選択図】 図1

認定・付加情報

特許出願の番号	特願 2002-171338
受付番号	50200853228
書類名	特許願
担当官	第七担当上席 0096
作成日	平成14年 6月13日

<認定情報・付加情報>

【提出日】 平成14年 6月12日

次頁無

特願 2002-171338

出 願 人 履 歴 情 報

識別番号

[000005821]

1. 変更年月日

1990年 8月28日

[変更理由]

新規登録

住 所

大阪府門真市大字門真1006番地

氏 名

松下電器産業株式会社